VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zálohování dat v síti pomocí open source nástrojů

Data Backup in Network with Open Source Projects

2013 **Jan Bonczek**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání bakalářské práce

Student: **Jan Bonczek**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R059 Mobilní technologie

Téma: Zálohování dat v sítí pomocí open source nástrojů
Data Backup in Network with Open Source Projects

Zásady pro vypracování:

Zálohování dat patří k velmi důležitým činnostem při správě nejen síťových serverů. Cílem bakalářské práce je navrhnout možnosti zálování dat v síti pomocí open source nástrojů.
1. Seznámení s problematikou zálohování dat v sítí.
2. Návrh možných řešení pomocí open source nástrojů.
3. Porovnání jednotlivých řešení, výhody a nevýhody.
4. Ověření navrženého řešení v laboratorních podmínkách.

Seznam doporučené odborné literatury:

Dorian Cougias, E. L. Heiberger, Karsten Koop, *The Backup Book: Disaster Recovery from Desktop to Data Center* Schaser-Vartan Books 2003, ISBN-13: 978-0972903905
W. Curtis Preston, *Backup & Recovery: Inexpensive Backup Solutions for Open Systems* O'Reilly Media 2007, ISBN-13: 978-0596102463
Podle pokynů vedoucího bakalářské práce.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Pavel Nevlud**

Datum zadání: 16.11.2012

Datum odevzdání: 07.05.2013

VYSOKÁ ŠKOLA BÁŇSKÁ - TECHNICKÁ UNIVERZITA * Fakulta elektrotechniky a informatiky * OSTRAVA *

prof. RNDr. Vladimír Vašinek, CSc.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Dne: *3. 5. 2013*

.....
podpis studenta

Poděkování

Rád bych poděkoval Ing. *Pavlu Nevludovi* za odbornou pomoc a konzultaci při vytváření této bakalářské práce.

Abstrakt

Bakalářská práce se zabývá možnostmi zálohováním dat v síti využitím open source nástrojů dostupných pod operačním systémem Linux. Zálohování patří mezi velice důležité činnosti, avšak mnohdy je opomíjeno nebo je prováděno v nedostatečném rozsahu či intervalu. Cílem této bakalářské práce je navrhnout možná řešení zálohování dat v síti pomocí zvolených open source nástrojů. V rámci této práce je také popsána obecná problematika zálohování dat, RAID pole a různé technologie pro ukládání dat.

Nejdůležitější částí této práce je vytvoření skriptů, které provádí zálohu pomocí zvolených nástrojů. Jako nástroje pro zálohování byly vybrány rsync, rdiff-backup, duplicity a hdup2. Navržená řešení jsou určena pro systémy s operačním systémem Linux a ověřována byla v laboratorních podmínkách na OS Ubuntu 12.04 LTS. Řešení jsou mezi sebou porovnána a popsány jejich výhody a nevýhody.

Klíčová slova

Zálohování, open source, Linux, rsync, rdiff-backup, duplicity, hdup2, RAID, inkrementální záloha, diferenciální záloha, plná záloha, DAS, NAS, SAN.

Abstract

Bachelor thesis deals with the possibility of backing up data in the networks with usage of open source tools which are available under the operation system Linux. Backing up is one of the most crucial processes, but most time is neglected or carried in small scale or span. The goal of this bachelor thesis is to come up with certain possibilities of the data backup in the network with the usage of selected open source tools. Part of this thesis deals with the general problematic of data backup, RAID fields and different technologies for storing data.

Crucial part of this thesis deals with creation of scripts, which are making backup with the usage of selected tools. Selected tools for backup are rsync, rdiff-backup, duplicity and hdup2. Suggested solutions are made for systems with operation system Linux and were tested in laboratory conditions on OS Ubuntu 12.04 LTS. Suggested solutions are compared between each other and theirs advantages and disadvantages are listed.

Key words

Backup, open source, Linux, rsync, rdiff-backup, duplicity, hdup2, RAID, incremental backup, differential backup, full backup, DAS, NAS, SAN.

Seznam použitých zkratek

Zkratka	Anglický význam	Český význam
ATA	Advanced Technology Attachment	Pokročilá technologie připojení
DAS	Direct-Attached Storage	Přímo připojené úložiště
DVD	Digital Versatile Disc	Digitální víceúčelový disk
FTP	File Transfer Protocol	Protokol pro přenos souborů
GNU	General Public License	Všeobecná veřejná licence
GPG	GNU Privacy Guard	Open source šifrovací systém
ID	IDentification	Identifikace
IETF	Internet Engineering Task Force	Komise techniky Internetu
LAN	Local Area Network	Lokální síť
PC	Personal Computer	Osobní počítač
PCI	Peripheral Component Interconnect	Sběrnice pro připojení periferii k základní desce
RAID	Redundant Array of Inexpensive/Independent Disks	Vícenásobné diskové pole laciných/nezávislých disků
RAM	Random-Access Memory	Paměť s přímým přístupem
SAS	Serial Attached SCSI	Sériově připojena SCSI sběrnice
SATA	Serial Advanced Technology Attachment	Seriál ATA
SCP	Secure Copy	Bezpečné kopírování
SCSI	Small Computer System Interface	Standardní rozhraní a sada příkazů pro výměnu dat
SSH	Secure Shell	Bezpečná příkazová řádka
TCP/IP	Transmission Control Protocol	Řídící přenosový protokol/ protokol Internetu
UNIVAC	UNIVersal Automatic Computer	Universální automatický počítač
URL	Uniform Resource Locator	Jednotný lokátor zdrojů
USB	Universal Serial Bus	Universální sériová sběrnice

Obsah

1 Úvod

Dnešní doba je charakteristická vzrůstajícím množstvím dat jak v domácnostech, tak ve firmách, z nichž mnohá jsou velice důležitá. V domácím prostředí to mohou být například rodinné fotografie, ve firmě se může jednat o důležité dokumenty, projekty, seznamy klientů a podobně. Pro uživatele mohou být tedy samotná data velice cenná a podle toho, jakou mají cenu, by jim měla být věnována patřičná míra zabezpečení. V nejhorších případech by jejich ztráta totiž mohla ohrozit i činnost firmy.

Počítačový hardware není naprosto spolehlivý, což je pádný důvod pro dodatečné zabezpečení dat, o která nechceme přijít. Selhání hardware však není jediný způsob, jak o data přijít. Krizové jsou také případy nechtěného smazání souboru, odcizení notebooku, živelných katastrof a dalších. I přestože jsou si uživatelé vědomi důležitosti svých dat, zálohování často patří mezi opomíjené činnosti, o něž se začnou blíže zajímat až v případě, že o data skutečně přijdou. Ani poté většinou nevyužijí sofistikované řešení, ale uchýlí se například k jednoduchému manuálnímu kopírování dat na jiné médium.

Bakalářská práce jak její název napovídá, se zabývá navržením možným řešením zálohování dat v síti využitím open source nástrojů dostupných v operačním systému Linux. Je rozdělena do tří hlavních částí, přičemž první část se zabývá obecnou problematikou zálohování dat. V druhé části je navrženo řešení zálohování pomocí open source nástrojů a obsahuje popis jejich výhod a nevýhod. Poslední část pojednává o laboratorním testování.

2 Seznámení s problematikou zálohování dat v síti

Cílem této práce je navrhnout řešení zálohování dat v síti využitím open source nástrojů. Současně si v teoretické části klade za cíl poukázat na důležitost vytváření záloh a dále chce také ukázat, že pro OS Linux jsou k dispozici kvalitní open source zálohovací nástroje, díky kterým lze navrhnout kvalitní zálohovací proces. V současnosti je tato důležitá činnost mnohdy opomíjena nebo není prováděná v dostatečném rozsahu, i když si uživatelé uvědomují její důležitost.

Tato kapitola se bude zabývat základními pojmy v oblasti zálohování dat v síti. Dále zde budou popsány jednotlivé metody, které lze využít pro zálohování dat a také význam RAID polí v oblasti zálohování dat v síti. Neméně důležitou částí bude také výběr toho, co se má zálohovat a použití vhodného média. Média pro zálohování jsou zmíněny pouze okrajově a větší prostor je věnován technologiím pro ukládání dat, jako jsou systémy DAS, NAS a SAN.

2.1 Význam zálohování dat

I přestože spolehlivost hardwaru je na vyšší úrovni, než tomu bylo v minulosti, neznamená to, že tím klesá význam zálohování dat; stále existuje velké množství možných příčin jejich ztráty. Data mají pro uživatele, kteří s nimi pracují, svou cenu, jež je dána mnoha aspekty. Samotná cena těchto dat by měla určovat míru zabezpečení proti případné ztrátě, protože ne všechna ztracená data lze obnovit jejich znovuvytvořením. Příkladem toho, kdy nelze získat ztracená data zpět, mohou být neopakovatelné vědecké experimenty či pozorování přírodních jevů.

2.2 Co je zálohování

Jednoduše by šlo říci, že zálohování je proces ochrany dat vytvořením jejich kopií. Tyto kopie poté mohou být využity při obnově ztracených dat. Je nutné si uvědomit, že zálohování má valný význam pouze tehdy, když je prováděno pravidelně, což zvlášť platí v počítačové síti. Nepravidelné zálohování v podobě ručního kopírování souborů není tím nejlepším řešením, protože je závislé na tom, kdy si uživatel vzpomene, že by měl svá data zálohovat. V případě velké počítačové sítě by se navíc mohlo jednat o proces velice náročný.

V této práci proto bude navrženo takové řešení zálohování pod operačním systémem Linux, aby bylo co nejvíce zautomatizováno. Kvalitní zálohovací systém je pro firmu, a tedy i pro počítačovou síť, základ, bez ohledu na její velikost. Naneštěstí si tento fakt velké množství firem neuvědomuje a neposkytují IT oddělení dostatečné prostředky k tomu, aby vytvořilo kvalitní zálohovací systém. Jednou z možností, jak i přesto vytvořit kvalitní zálohovací systém, je využití open source nástrojů, určených pro OS Linux. Tím se ušetří prostředky za nákup licencí na placené programy určené pro zálohování dat.[2]

2.3 Jaká data zálohovat

Nejlepší a zároveň nejsnadnější možností by bylo zálohovat celý server se všemi daty a nainstalovanými programy. To by ale nemuselo být vhodné řešení, protože v systému se určitě nachází data, která není nutné zálohovat. Došlo by k zbytečnému zvětšení nároků na kapacitu zálohovacího média, ale také ke zvýšení zátěže sítě v případě, že by se zálohovalo vzdáleně.

Možností, jak určit data, která chceme zálohovat, je definovat si přijatelné ztráty ještě před samotným vytvořením zálohovacího plánu [2]. Měla by být tedy provedena důkladná analýza systému, která by určila data, jež je nutné zálohovat. To také rozhodne o tom, kolik peněz investovat do diskové kapacity určené pro ukládání zálohy.

Příkladem může být zálohování domovského adresáře sekretářky, který by obsahoval velké video soubory (například 700 MB). Přijatelnou ztrátou by bylo, kdyby se jednalo například o filmy, které si sekretářka pouští. V tom případě by bylo zbytečné zálohovat tyto soubory a mohly by se ze

zálohování vynechat, aby nezabíraly zbytečný prostor. Ovšem nemusí to být pravidlo, proto je třeba právě analýza systému.

Mezi typické typy dat, které by měly být zahrnuty do zálohy, patří dle [1] níže zmíněné:

Co zálohovat

- Domovské adresáře uživatelů. Typická cesta je /home/
- Systémové konfigurační soubory nacházející se v /etc/
- Obsah webových stránek. Typická cesta je /var/www/
- Pošta uživatelů nacházející se v /var/mail/
- Seznam nainstalovaného software
- Databáze
- A další

Co nezálohovat

- Nainstalované aplikace, stačí zálohovat konfigurační soubory
- /proc/ je tvořen automaticky systémem
- Dále potom můžeme vynechat některé typy souborů (.avi, .exe, .jpg,)

Abychom měli jistotu, že jsme do zálohy zahrnuli vše potřebné pro případnou obnovu dat po havárii, je vhodné otestovat „nanečisto“ obnovu například na virtuálním stroji.[1]

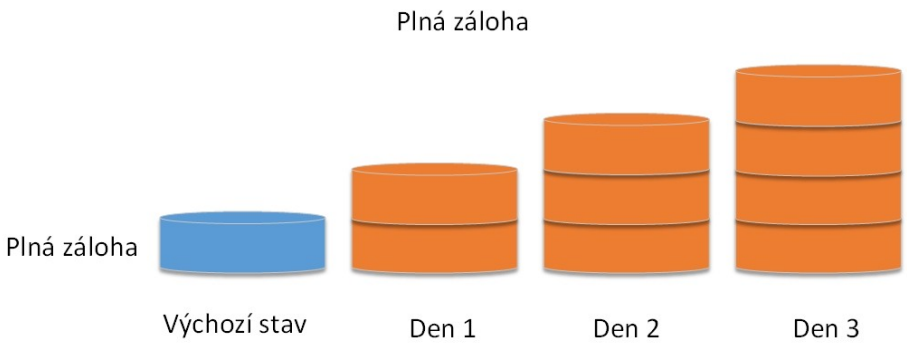
Testováním předejdeme tomu, abychom během obnovy zjistili, že nejsme schopni systém obnovit do původního stavu, protože jsme zapomněli do zálohy zahrnout vše důležité. Nezahrnutí potřebných dat do zálohy je podle [2] jedna z častých možností, jak o ně přijít i v případě pravidelného zálohování.

2.4 Metody zálohování

Existují tři základní metody pro zálohování dat. Jedná se o plné zálohování, inkrementální zálohování a diferenciální zálohování, přičemž každá z těchto metod má své výhody i nevýhody a jsou různě náročné na kapacitu média, na které se zálohuje. Jednotlivé typy vynikají v různých aspektech, ať už se jedná o rychlost vytvoření zálohy, jednoduchost obnovy dat ze zálohy, jednoduchost vytvoření zálohy a další. Tou nejčastěji využívanou metodou je inkrementální zálohování.

2.4.1 Plná záloha

Je takový typ zálohování, kdy se zálohují všechna data, která jsme zahrnuli do zálohy, bez ohledu na to jestli se od poslední plné zálohy změnila nebo ne. Tento typ zálohování s sebou přináší jak určité výhody, ale tak i nevýhody. Výhodou je to, že obnovení ze zálohy je jednoduché. V případě, že chceme získat pět dní stará data, provedeme obnovu z pět dní staré zálohy, pokud je k dispozici. Nevýhodou tohoto řešení je ovšem jeho náročnost na kapacitu zálohovacího média, jelikož jsou zálohována i ta data, která se od poslední zálohy nezměnila, a jejich obnovení by bylo možné z jiné starší zálohy. Princip plné zálohy znázorňuje obrázek 2.1.

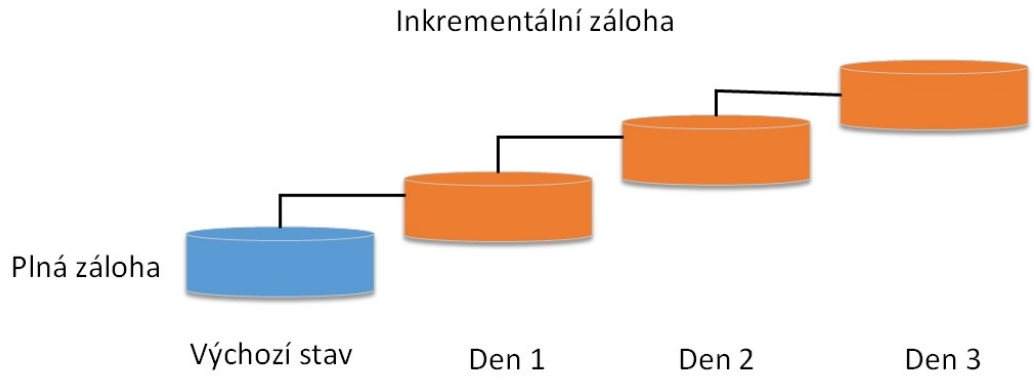


Obrázek 2.1: Princip plné zálohy

2.4.2 Inkrementální záloha

Patří mezi nejpoužívanější metodu zálohování, protože vyniká nejlepšími vlastnostmi jak v nízké náročnosti na kapacitu zálohovacího média, tak v rychlosti vytvoření zálohy. Inkrementální záloha začíná vytvořením plné zálohy, ostatní zálohy už obsahují pouze ty soubory, které se změnily od poslední plné nebo inkrementální zálohy. Inkrementální zálohování má menší nároky na kapacitu zálohovacího média, protože nejsou vytvářeny duplicitní kopie stejných dat. Rovněž je snížen síťový provoz a čas potřebný pro vytvoření zálohy. Princip inkrementální zálohy znázorňuje obrázek 2.2.

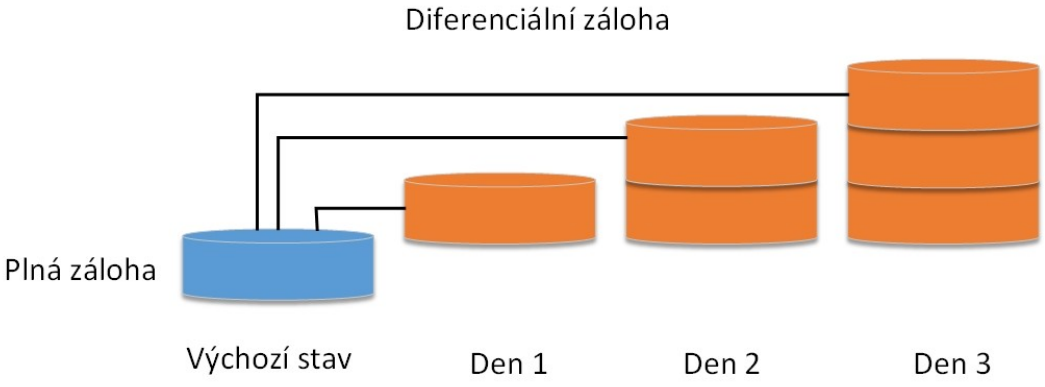
Inkrementální zálohování je složitější v případě obnovy ze zálohy. Jelikož jsou na sebe jednotlivé inkrementální zálohy vzájemné navázány, je v případě obnovy nutné mít všechny předchozí inkrementální zálohy a původní plnou zálohu.



Obrázek 2.2: Princip inkrementální zálohy

2.4.3 Diferenciální záloha

Je také vytvářena vůči plné záloze, ale na rozdíl od inkrementální zálohy na sobě nejsou jednotlivé diferenciální zálohy závislé. To znamená, že v případě selhání je obnova dat snadnější a rychlejší. Jelikož se zálohuje vše, co se změnilo od plné zálohy, a ne pouze to, co se změnilo od poslední jakékoliv zálohy jako v případě inkrementální, tak postupně narůstá objem dat, která je třeba zálohovat, a tím i síťový provoz a čas potřebný pro vytvoření zálohy. Princip plné zálohy znázorňuje obrázek 2.3.



Obrázek 2.3: Princip diferenciální zálohy

2.5 Média pro zálohování dat

Je-li řeč o zálohování dat v síti, mluvíme nejčastěji o dvou typech médií. Jedná se o pevné disky a o magnetické pásky. Ostatní média jako jsou CD, DVD nebo USB flash disky nebudou

podrobněji popisována. Pří výběru vhodného média je důležité zvážit náklady, rychlost, spolehlivost, dostupnost, kapacitu a použitelnost [3].

Spolehlivost je v případě zálohování klíčová. Data, která zálohujeme, by měla vydržet bez toho, aby došlo k jejich poškození, i několik let. Na spolehlivosti se také podílí to, jak médium používáme. I velice drahý a spolehlivý disk může být k ničemu v okamžiku, když je ve stejné místnosti jako disk, ze kterého zálohujeme. [3]

Na kapacitě bude záležet to, kolik starých záloh může být ponecháno, než se začnou mazat (archivovat) staré zálohy. Celková velikost zálohy totiž bude neustále narůstat, proto je třeba úložiště čistit od starých záloh a uvolňovat tak místo pro zálohy nové a aktuálnější.

Dalším důležitým aspektem je dostupnost. Jestliže již je vyřešena automatizace, je velice důležité zajistit dostupnost média, na které se zálohuje. Nelze totiž zálohovat v případě, kdy je médium nedostupné. [3]

O zvýšení dostupnosti bude více zmíněno v kapitole o diskových polích RAID 2.6.

2.5.1 **Magnetické pásky**

První použití magnetické pásky bylo už v roce 1951 v počítači UNIVAC. Magnetické pásky si ponechávají svou oblíbenost v oblasti ukládání dat. Jsou výhodnější v poměru kapacita/cena v porovnání s pevným diskem. Svoji oblíbenost si rovněž zachovaly, protože dokážou data udržet bez poškození po velice dlouhou dobu (až 30. let) a jsou snadno přenositelné. [12]

Magnetické pásky jsou tedy stále využívány v oblasti zálohování dat. V praxi se využijí například pro archivaci záloh, kdy aktuální zálohy jsou na rychlých pevných discích a starší zálohy na magnetických páskách.

2.5.2 **Pevný disk**

Vzhledem k tomu, že je v současnosti kladen velký důraz na rychlou obnovu ze zálohy, nemusí být vždy výhodné zálohovat pouze na pásková média. Možnost jak snížit čas, po který je systém po havárii nefunkční, je využít kvalitních a vysokootáčkových pevných disků. V případě, že je záloha prováděna po síti, je po dobu provádění této zálohy část kapacity sítě využita pro přenos zálohy a ta bude opět rychlejší, pokud použijeme vysokootáčkové pevné disky v kombinaci s RAID poli.

Hlavní výhodou pevných disků jsou nízké přístupové doby, dostupnost, kapacita a snadné použití. Nevýhodou je jejich snadné poškození, zejména při přepravě, a jejich schopnost dlouhodobého uložení dat je relativně neznámá. A také samotná přenositelnost není tak jednoduchá jako v případě magnetické pásky, kterou stačí vyjmout z mechaniky. [5]

2.6 **RAID**

Jak již bylo zmíněno, důležitým aspektem při tvorbě záloh je dostupnost. Jedním ze způsobů jak zvýšit dostupnost je využití RAID polí. Nelze zcela ovlivnit to, jak rychle se budou HDD poškozovat a kdy by mohlo dojít k případnému selhání a tím ke ztrátě dat nebo nemožnosti data zálohovat nebo obnovovat.

RAID je metoda, která kombinuje několik fyzických disků do jednoho pole, pro zvýšení rychlosti zápisu a čtení (RAID 0), pro odolnost vůči chybám (RAID 1) nebo pro kombinaci obojího (například RAID 5). V systému se poté takovéto diskové pole jeví jako jedna logická jednotka nebo disk. Základní strukturou RAID je tedy pole: skupina disků uspořádaná takovým způsobem, aby pracovala efektivněji. Systémy RAID, které zajišťují odolnost proti výpadku disku, jsou velice často označovány jako „fault tolerance“, tedy odolné vůči chybám.

RAID, který je odolný vůči chybám, sice zvyšuje provozuschopnost systému tím, že poskytuje přístup k datům i v těch případech, kdy některý z disků selže. Je ovšem velice důležité si uvědomit, že RAID není zálohování. RAID, který je odolný vůči chybám, sice svými vlastnostmi může připomínat jednoduchou zálohu, ale nechrání data proti všem typům ztráty. Dokáže ochránit systém při selhání

disku, ale už ne v případě, že uživatel data smaže nebo přepíše. Změny jsou promítnuty na všechny disky v poli a původní data jsou ztracena jako v případě použití jednoho disku. RAID neochrání data v případě požáru a dalších živelných katastrof.

Z pohledu funkcionality lze rozlišit dva základní typy RAID, jedná se o hardwarový a softwarový RAID.

Hardwarový RAID je speciální hardwarová součást systému, která umožňuje bez instalace dodatečných ovladačů spravovat disková pole. Často výrobci základních desek označují, že obsahuje RAID, ale ve většině případu se jedná o falešný RAID a ne o skutečný hardwarový, jak by se mohlo zdát.

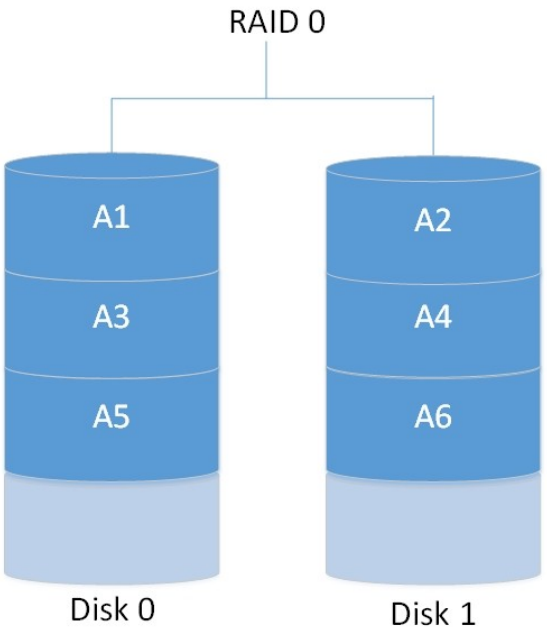
Skutečný hardwarový RAID má totiž vlastní procesor, který se stará o výpočty paritních informací a také má vlastní paměť. Je zcela nezávislý na operačním systému. Dále zajišťuje rozložení výkonu v rámci typu RAID na pevné disky, stará se o sestavení, obnovu a připojování disků. Cena skutečného hardwarového RAID je mnohem vyšší než cena toho falešného. [13]

Softwarový RAID nemá vlastní procesor a pro výpočet paritních informací využívá procesor serveru, čím zabírá procesorový výkon. Softwarový RAID je vhodný pro nasazení těch typů RAID, u kterých není vypočítávána parita. V případě RAID, u kterého je vypočítávána parita, záleží na procesorovém výkonu daného systému. RAID nemusí znatelně snížit výkon serveru v případě moderních výkonných procesorů.

V operačním systému linux se o sestavení softwarového RAID stará nástroj mdadm. Ve verzi systemu Linux Ubuntu 3.2.0-30-generic-pae a verzi mdadm - v3.2.5 jsou k dispozici následující režimy: RAID 0, RAID 1, RAID 4, RAID 5, RAID 6 a RAID 10.

2.6.1 RAID 0

RAID 0 neboli prokládaný svazek. Jeho nasazení přináší zvýšení rychlosti operací čtení a zápisu, jelikož jsou zapisované soubory rozděleny na menší bloky, které jsou ukládány střídavě na všechny disky v poli. Zapojení tohoto typu RAID nepřináší žádné snížení kapacity sestaveného pole. Jestliže by bylo pole složené ze dvou disků 120 GB a 320 GB, je výsledná kapacita pole limitovaná kapacitou menšího disku, tedy 2 * 120 GB. RAID 0 není odolný vůči chybám a při selhání jednoho disku jsou ztracena všechna data. Minimální počet disků v poli je 2. Princip znázorňuje obrázek 2.4.



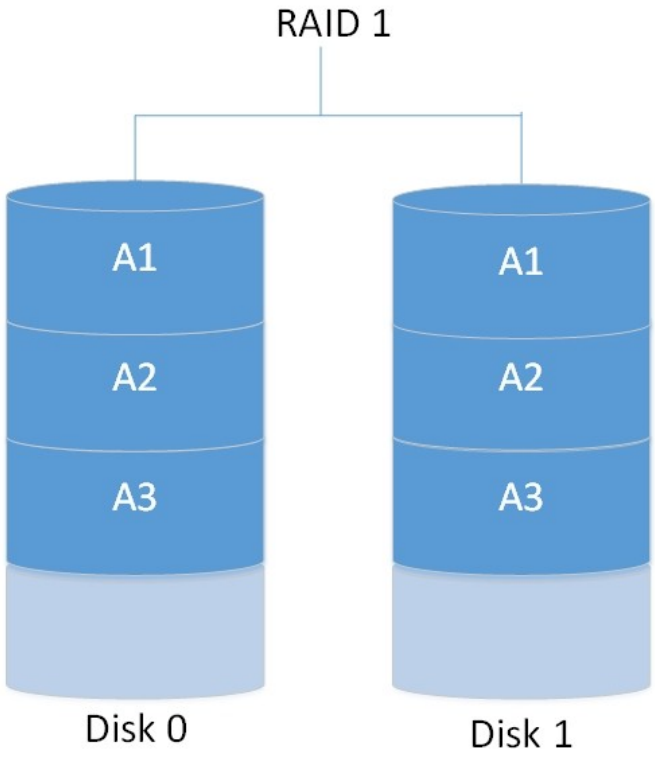
Obrázek 2.4: Ukládání dat na pole RAID 0

2.6.2 **RAID 1**

Je první RAID, který je odolný vůči chybám. V poli musí být zapojeny minimálně 2 disky opět nejlépe se stejnou kapacitou (jinak je výsledné pole rovno kapacitě menšího disku). RAID 1 je také nazýván zrcadlení, jelikož zapisovaná data jsou ukládána zároveň na více disků, princip znázorňuje obrázek 2.5. V případě použití minimálního počtu disků v poli může dojít bez ztráty dat k selhání až jednoho disku v poli, v případě kdy jsou v poli 3 aktivní disky, může dojít k selhání až dvou disků. V takovém případě jsou data okamžitě přístupná z druhého disku. Navíc může být RAID 1 doplněn o záložní disky.

Operace čtení je rychlejší než v případě využití jednoho disku, protože se zároveň čte z více disků najednou. Zápis je oproti tomu pomalejší, jelikož se zapisuje na více disků najednou.

U velkých RAID 1 polí může důsledkem zrcadlení dojít k tomu, že se zahltí PCI sběrnice. V tomto případě má lepší vlastnosti hardwarový RAID, u kterého data prochází přímo řadičem a ne přes PCI sběrnici. [3]



Obrázek 2.5: Ukládání dat na pole RAID 1

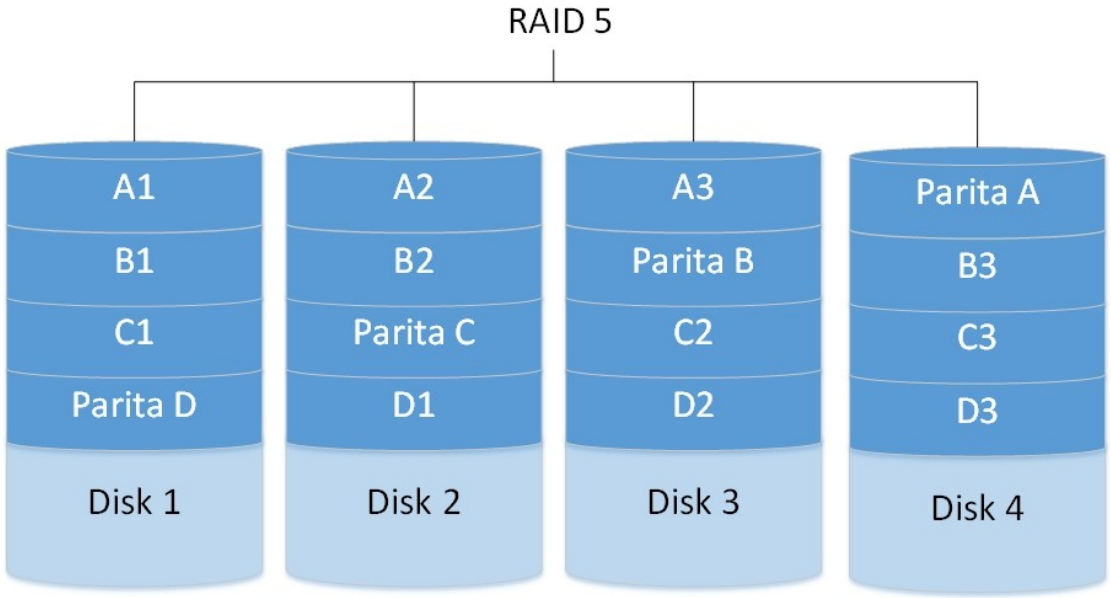
2.6.3 **RAID 4**

Tento typ zapojení není příliš využíván. RAID 4 lze zapojit na minimálně třech discích, přičemž na jeden z nich jsou ukládány pouze paritní informace. Paritní informace mohou být v případě selhání jednoho disku využity pro obnovu ztracených dat, v případě selhání dvou disku dochází ke ztrátě dat. Důvod proč se příliš nevyžívá, je právě v tom, že disk na který jsou ukládány paritní informace se stává úzkým hrdlem systému. Použití si najde v případech více pomalejších disků a jednoho rychlejšího, který slouží pro ukládání parity. Záložní disky jsou podporovány. [3]

2.6.4 **RAID 5**

Tento typ pole musí obsahovat alespoň 3 aktivní disky o stejné velikosti, které mohou být rovněž doplněny o záložní disky. Data jsou zapisována střídavě na všechny disky v poli, obdobně jako u RAID 0, ale tento typ RAID je již odolný vůči chybám, jelikož si navíc zaznamenává paritní informaci. Na rozdíl od RAID 4 je paritní informace střídavě zapisována na všechny disky v poli a dohromady zabírá kapacitu jednoho disku. Parita může být využitá pro případnou rekonstrukci dat. Princip znázorňuje obrázek 2.6.

Při použití minimálního počtu disků v poli a při případném selhání jednoho disku jsou chybějící data dopočítána z paritních informací. Při výpadku dvou disků jsou všechna data ztracena. Po dobu, kdy pole běží v degradovaném režimu (pole je nekompletní vinou selhání disku), je snížen jeho výkon. Čtení z RAID 5 je stejně rychlé jako u RAID 0, ale zápis může být vzhledem k tomu, že se musí vypočítávat paritní informace, pomalejší [3].

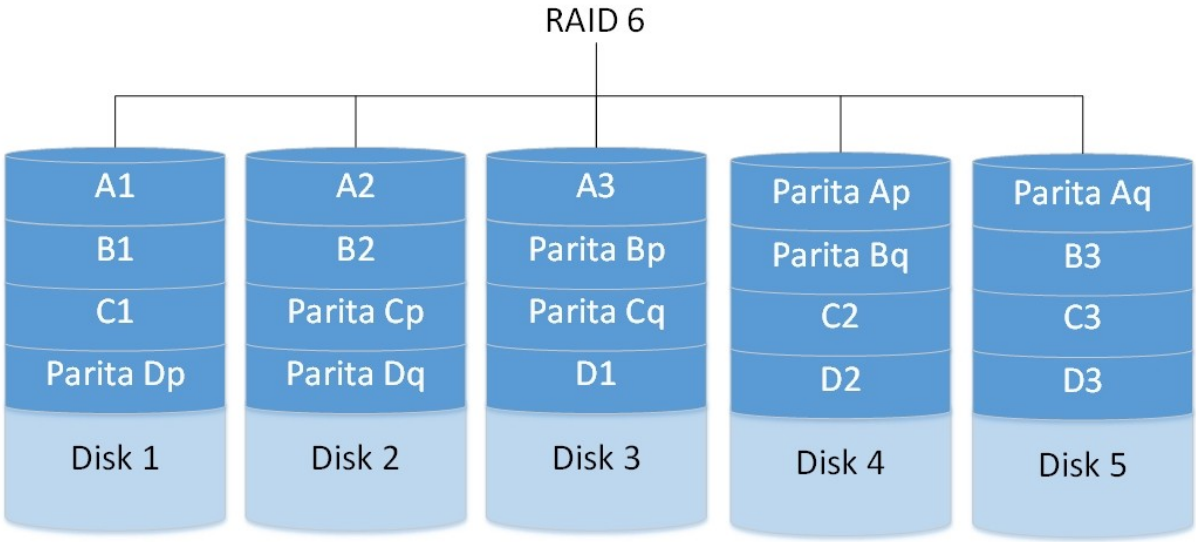


Obrázek 2.6: Ukládání dat na pole RAID 5 [14]

2.6.5 **RAID 6**

Jedná se o obdobu RAID 5 s tím rozdílem, že se vytváří 2 paritní informace a každá z nich je vypočítána jiným způsobem. Pro vytvoření pole jsou potřeba minimálně 4 disky s jedním nebo více záložními disky. Paritní informace jsou střídavě zapisovány na všechny disky v poli a zabírají kapacitu dvou disků. Takovéto pole je poté odolné vůči výpadku až dvou disků v poli. Princip znázorňuje obrázek 2.7.

RAID 6 je sice dražším řešením, ale vhodné zejména tam, kde mají jednotlivé disky velkou kapacitu. Když dojde k selhání disku u pole RAID 5, je poté zranitelné po dobu, než je disk nahrazen a data se zrekonstruují. Právě v případě použití disků o velké kapacitě může tato operace trvat mnoho hodin. Řešením je tedy RAID 6, který zabezpečí ochranu dat i při selhání jednoho disku a po dobu, než se zrekonstruují data. Rychlost čtení je srovnatelná s RAID 5, ale zápis je opět o něco pomalejší, protože se musí vypočítávat 2 paritní informace.

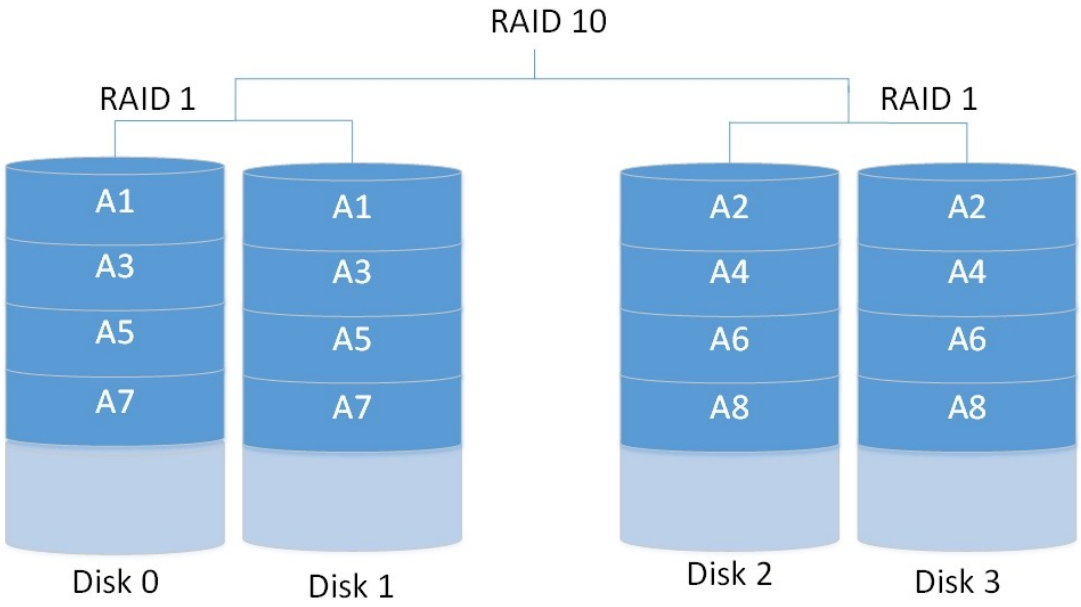


Obrázek 2.7: Ukládání dat na pole RAID [14]

2.6.6 **RAID 10**

Jedná se kombinaci RAID 1 a RAID 0. Jde o pole v poli, kdy se vytvoří 2 zrcadlená pole a nad nimi se vytvoří jedno pole RAID 0. Výhodou tohoto řešení je vysoký výkon a odolnost, protože odpadá potřeba vypočítávat paritní informace. Princip znázorňuje obrázek 2.8.

Je odolný proti havárii až jednoho disku v každém sub-poli.



Obrázek 2.8: Ukládání dat na pole RAID 10

2.6.7 **Záložní disk**

Záložní disk je takový disk, který je připojen do RAID pole, ale je neaktivní. Jeho stav se z neaktivního na aktivní změní v okamžiku, kdy je detekováno selhání některého aktivního disku v RAID poli.

Jako příklad lze uvést RAID 1, kdy jsou použity 2 disky a selže-li jeden z nich, dojde k okamžitému započetí synchronizace. Jedná se tedy o dodatečnou ochranu. Další výhodou je, že synchronizace začne okamžitě, takže v případech, kdy není možné okamžitě vyměnit poškozený disk v poli ručně za nový a systém již není chráněn proti selhání disku, zkrátíme čas, po který běží systém v degradovaném režimu.

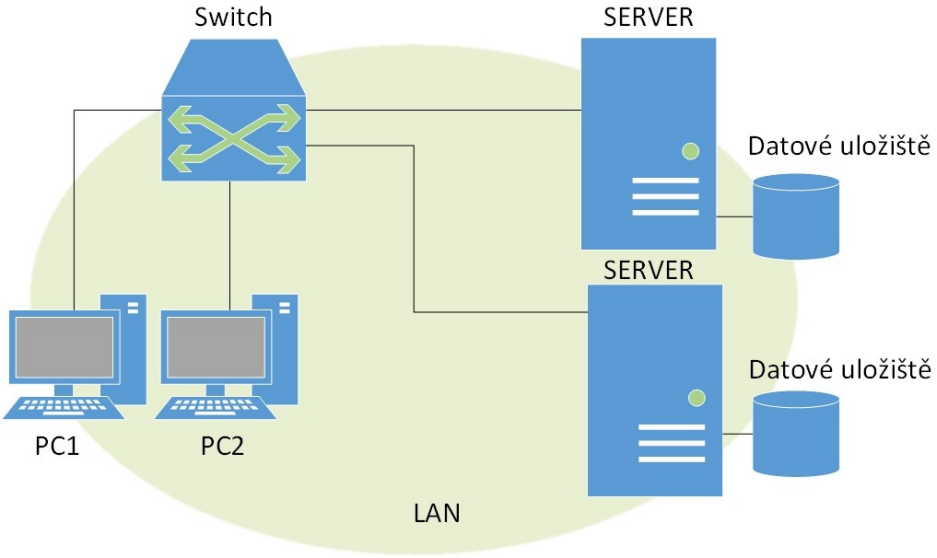
2.7 Technologie pro ukládání dat v síti

Pod pojmem technologie pro ukládání dat, nebo také architektury úložišť se rozumí způsoby, kterými lze mít připojené datové úložiště k PC či serveru. Mezi nejběžnější technologie patří DAS, NAS a SAN. Výběr vhodné technologie by měl být dobře zvážen s ohledem na budoucí požadavky. Dále budou popsány technologie DAS, NAS a SAN.

2.7.1 DAS

Je nejběžnější způsob připojení disků nebo rozšiřujících diskových polí ke PC nebo serveru. Disky jsou připojeny přímo kabelem (mezi serverem a diskem se nenachází žádný síťový prvek jako switch nebo router). Nejběžnějšími používanými disky jsou například ATA, SATA, SCSI, SAS, ale jedna se i o externí disky připojované pomocí USB či firewire. [6]

Výhodou jsou nízké počáteční náklady a jednoduchost zapojení. Nevýhodou je, že v případě výpadku serveru ke, kterému je disk připojen, nemůže s daty manipulovat jiný server. Z toho plyne, že vytváření záloh je na těchto systémech obtížnější. Další nevýhodou je rozšiřitelnost kapacity. V případě, že nám již nestačí kapacita stávajícího diskového pole, je potřeba koupit nové diskové pole, což může být spojeno s náklady až desítek tisíc korun.



Obrázek 2.9: Datové úložiště typu DAS

2.7.2 NAS

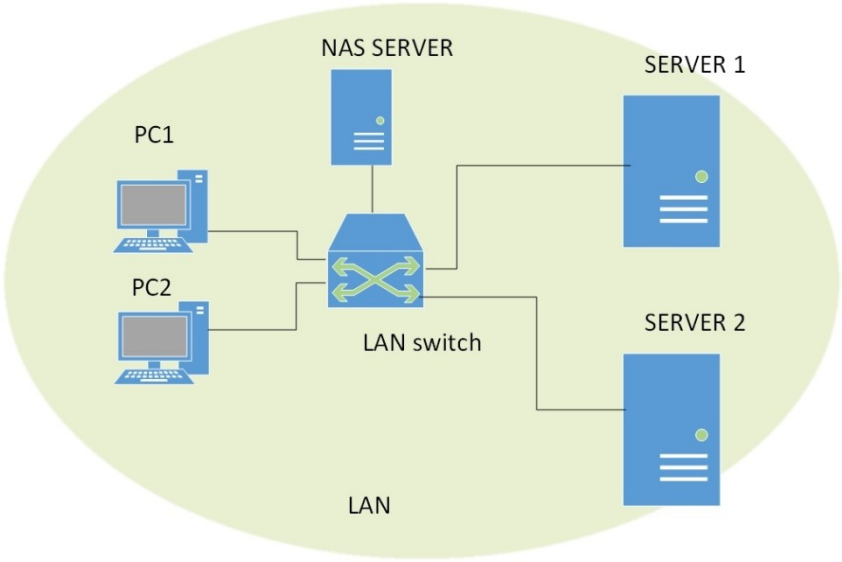
Oproti DAS je NAS přímo uzpůsoben ke sdílení dat na úrovni souborů v síti LAN. Jedná se většinou o speciální počítač postavený od základu pro sdílení dat [7]. Lze na něj nainstalovat různé operační systémy - jak Windows, tak i Linuxové open source distribuce určené pro NAS servery, jako například FreeNAS, NASLite a další, ale lze také nainstalovat běžný systém jako debian či ubuntu. V praxi bývá OS spouštěn z externího USB zařízení v podobě flash paměti.

Obsahují 1 a více za chodu odpojitelných a připojitelných pevných disků, které mohou vytvářet RAID pole pro zvýšení dostupnosti. Jeden NAS může obsluhovat jak stanice s operačním systémem Windows, tak i Linuxové. Jedná se vlastně o známé připojení síťového disku pomocí

protokolu SMB u Windows a NFS pro Linux. Pro klienta se potom takto připojený disk jeví jako přímo připojený a pracuje s ním jako s klasickým diskem a neví nic o souborovém systému, který je na NAS vytvořen.

Hlavním rozdílem NAS oproti DAS je jednoduché rozšíření diskové kapacity serveru prostým připojením NAS do stávající LAN sítě, protože nemají žádné zvláštní požadavky na již vybudovanou síť. NAS servery jsou jednoduché na obsluhu, protože jsou důležitá data uložena na jednom místě a tím pádem poskytují snadnější možnosti zálohování (zálohujeme vše z jednoho místa a ne několik serverů s vlastními disky).

NAS se stávají čím dál oblíbenější jak v domácnostech pro ukládání a snadné sdílení multimediálních dat, tak také ve firmách pro svou snadnou správu, možnosti snadného zálohování a dnes již i levné pořizovací náklady. Můžou být také využity i k jiným účelům než je samotné sdílení dat, například jako jednoduchý www server, ale nejsou k tomuto účelu primárně stavěny. Cena NAS serveru se samozřejmě odvíjí od toho, kolik může mít připojených disků nebo jaké typy RAID podporuje. Určitá nevýhoda při použití NAS plyne z toho, že je připojen na stávající LAN síť a v případě mnoha požadavků může síť zatěžovat. [7]

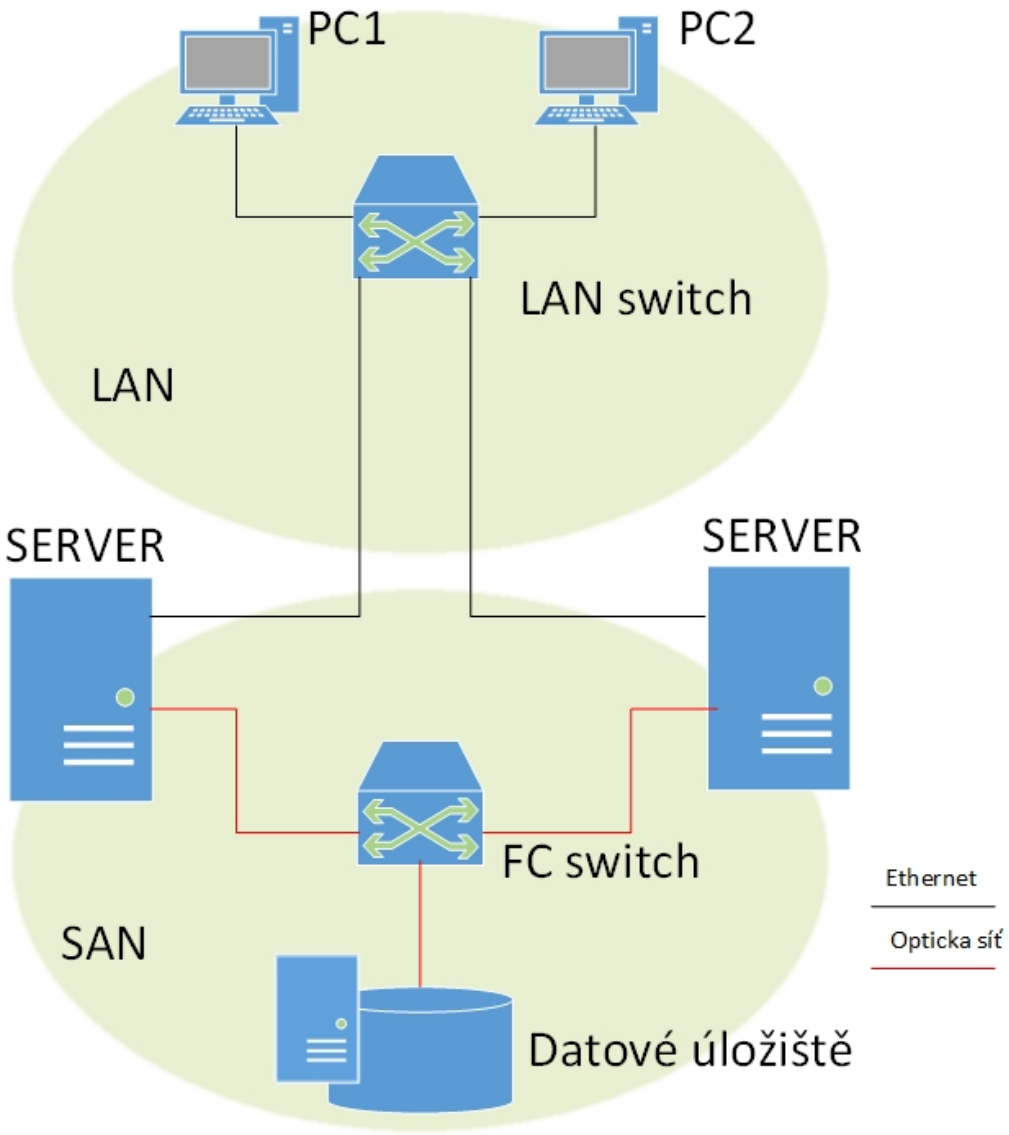


Obrázek 2.10: Datové úložiště typu NAS

2.7.3 SAN

Je zcela oddělená vysokorychlostní datová síť (nezávislá na LAN), která slouží k připojení externích zařízení k serveru. Jedná se hlavně o disková pole a páskové knihovny. Využívá Fibre Channel (dále FC), což je komunikační rozhraní používané pro vysokorychlostní přenos dat. SAN umožňuje oproti NAS sdílení dat na úrovni bloků. Pracuje se surovými daty a nemá pojem o souborovém systému, který je nad ním vybudován. Souborový systém vytvoří až samotný server, kterému se disk jeví jako nenaformátovaný a přímo připojený. Použití SAN s sebou přináší obrovské pořizovací náklady (FC switch až stovky tisíc korun, většinou je potřeba pořídit minimálně 2 a dále pak FC adaptery pro servery), proto je využívána především ve velkých organizacích jako jsou banky nebo automobilový průmysl. [8]

K propojení serverů se samotným diskovým úložištěm se používá optický kabel a speciální prvky jako FC switche a FC adaptéry u serverů. Jelikož je k propojení použita optická technologie, mohou být data přenášena na velké vzdálenosti desítek, stovek i tisíců kilometrů v závislosti na použité technologii.



Obrázek 2.11: Datové úložiště typu SAN [9]

Na obrázku 2.11 je znázorněna LAN síť (černé spojení) a od ní oddělená samostatná SAN síť (červené spojení) s diskovým úložištěm, ke kterému jsou připojeny dva servery. Každému serveru je kapacita přidělována dynamicky. Toto řešení se už blíží k řešení vysoké dostupnosti, ve kterém by byl jeden server v případě havárie schopný zaujmout místo po havarovaném serveru, protože diskový prostor může byt zpřístupněn náhradnímu hardware. Aby to bylo možné, je potřeba do serveru přidat navíc FC kartu, což by chránilo server proti poruše jedné FC karty, dále zdvojit switch (proti poruše switche), každý switch připojit k samostatnému řadiči diskového pole (ty nejlepší diskové pole mají redundantní řadiče RAID) a vše připojit k různým zdrojům nepřetržitého napájení. [9]

iSCSIS je protokol který umožňuje přenášet příkazy SCSI v běžných TCP/IP sítích a připojovat úložná zařízení. Velkou roli hrál pro iSCSI rok 2003, kdy ho organizace IETF uznala jako standardní protokol rodiny TCP/IP.

První verzi iSCSI představila firma Intel v roce 2001. Jedná se o přímého konkurenta již zavedené technologie FC. iSCSI vychází ze známých technologií, tím prvním je SCSI, který slouží pro připojování disků v serverech, a tím druhým je protokol TCP/IP, který využívá většina počítačových

sítí. Komunikace se pak odehrává mezi tzv. iniciátorem (server, PC stanice) a SCSI target (jeden a vice ISCI disků) a opět se takto připojené disky tváří, jako kdyby byly přímo připojené. [19]

Přináší významné úspory financí, protože není potřeba kupovat speciální FC prvky, ale vystačí si s již zavedenou technologií.

2.8 Práce s poškozenými médii

Mohlo by se zdát, že v případě, kdy dojde k havárii například pevného disku, tak z něj již nelze získat žádná data. Jestliže by došlo jen k poškození čtecího nebo zapisovacího mechanismu, pak je sice disk nepoužitelný pro ukládání záloh nebo čtení souborů, ale samotná data nemusí být poškozena. V praxi se proto setkáváme s likvidací pevných disků, které obsahují citlivá data.

Jestliže se tedy na disku nachází opravdu citlivá data, je nutné zamezit jejich získání. Lze provést softwarové smazání dat, to je ovšem mnohdy zdlouhavé (ne vždy možné), a proto je jednodušší cestou pevný disk rozebrat, oddělit samotnou elektroniku od ploten a plotny poté zničit tak, aby z nich nebylo možné už nic získat.

3 Návrh možných řešení zálohování dat pomocí open source nástrojů

V open source podobě je pro OS Linux šířeno velké množství zálohovacích nástrojů. Využití těchto nástrojů přináší snížení celkových nákladů na proces zálohování. Pro návrh možných řešení zálohování dat jsem zvolil čtyři nástroje. Jedná se o rsync, rdiff-backup, duplicity a hdup2, které budou dále popsány a pro každý z nich je vytvořen zálohovací skript, tedy řešení zálohování dat. Největší pozornost je věnována rsync, který tvoří základ pro mnoho dalších nástrojů. V průběhu této kapitoly tedy představím vybrané nástroje, popíši základní práci těmito nástroji, dále představím navržené řešení a jejich výhody a nevýhody.

Ve všech případech se bude jednat o inkrementální typ zálohování, protože tento typ zálohování je nejčastěji využíván. Oblíbenost inkrementální zálohy je dána tím, že u ostatních typů záloh dochází k rychlému nárůstu velikosti zálohy a tedy k zvýšeným nárokům na kapacitu zálohovacího média. Navržená řešení se zabývají zálohováním linuxových systémů a jako médium pro ukládání záloh může být využit pevný disk či USB flash disk.

3.1 Automatické spouštění skriptů

Jedna z podmínek vytvoření kvalitního zálohování je splnění podmínky, aby zálohování bylo spouštěno automaticky. K tomu může být využit nástroj cron.

Cron je Linux/Unix nástroj, určený pro plánované spouštění procesů (skriptů) v předem definovaném čase a intervalu. Nelze pomocí něj spouštět skripty či příkazy jednorázově, ale pouze v určitých intervalech. Zároveň je nutné mít skripty navržené tak, aby nebyly interaktivní (například zadávání hesla).

Tabulku cron otevřeme zadáním crontab –e, tato tabulka obsahuje ukázku zadávání. Pokud chceme, zajistit opakované spouštění skriptu, provádí se nastavení cronu v následujícím formátu.

```
minuta hodina den_v_měsící měsíc den_v_týdnu cesta_ke_skriptu
```

Den v týdnu se zadává v intervalu 0-6 (0 je neděle), měsíc v intervalu 1-12. Hvězdička znamená libovolnou hodnotu.

Ukázky zadávání:

Následující řádek spustí skript rsync.sh každý den ve 3 hodiny ráno.

```
0 3 * * * /root/bin/rsync.sh
```

Další ukázka spouští skript ve 3 hodiny ráno každou minutu až do 3:59.

```
* 3 * * * /root/bin/rsync.sh
```

Poslední ukázka spouští skript ve 3:15, každé pondělí.

```
15 3 * * 1 /root/bin/rsync.sh
```

Možnosti jak pracovat se zadáváním intervalů pro cron je více viz [15]. Pro účely spouštění zálohovacích skriptů, by měly stačit zmíněné ukázky.

3.2 Řešení zálohování nástrojem rsync

Popis nástroje

RSYNC patří mezi velice často používané zálohovací nástroje v linuxových systémech vůbec. Umožňuje vytvářet jednoduché, ale účinné inkrementální zálohy jak na lokální disk, tak na vzdálený disk využitím protokolu SSH, který přenáší data po sítí v zašifrované podobě. Dostupný je pod licencí

GNU a většinou již bývá součástí instalace systému. Nabízí velké množství možností, pomocí kterých lze kontrolovat a velmi přesně specifikovat soubory určené k zálohování. Využívá delta-transfer algoritmus, díky kterému lze snížit množství dat přenášených přes síť tím, že se posílají jenom rozdíly mezi zdrojem a cílem. Nepodporuje zálohování mezi jedním vzdáleným počítačem a jiným vzdáleným počítačem. [10]

O jeho kvalitách svědčí i to, že mnohé z dalších nástrojů pro zálohování jsou založeny a využívají právě rsync, a pouze zjednodušují jeho používání. Jako příklad může být zmíněn Grsync, což je grafická nástavba rsync, a dále pak také rdiff-backup či duplicity, které usnadňují vytváření inkrementálních záloh a mnoho dalších nástrojů.

Základní práce

Nejjednodušší použití rsync je při lokální záloze. Jako první jsou zvoleny přepínače, poté adresář, který zálohujeme (dále jen zdrojový adresář) a kam chceme zálohovat (dále jen cílový adresář). Následující příklad vytvoří zálohu adresáře student do adresáře backup.

```
rsync -avh /home/student/ /media/backup/
```

Ve výše zmíněném příkazu přepínač `-a` zajistí, že rsync pracuje v archivačním módu, to znamená, že zachovává oprávnění, časy vytvoření, vlastníka a skupiny. Druhý přepínač `-v` zobrazuje vypisování informací během zálohování, množství zobrazovaných informací lze ovlivnit zadáním až `-vvv`, `-h` převádí údaj o množství přenesených dat z byte na lépe čitelnou jednotku (MB, GB atd.).

Další možností, jak již bylo zmíněno, je přenášet zálohy vzdáleně s využitím SSH, které zajistí šifrovaný přenos. Přibude tedy přepínač `-e`, díky kterému lze specifikovat protokol, který se použije pro přenos. V případě, že by byl přepínač `-e` vynechán, je automaticky použito SSH. Vzdálený stroj, na který zálohujeme, je poté specifikován uživatelským jménem a IP adresou - viz následující ukázka.

```
rsync -avhe ssh /home/student/ user@IP:/media/backup/
```

Průběh přenosu dat lze zobrazit doplněním přepínače `--progress`. Tento přepínač zobrazí průběh synchronizace jako je přibližný čas a rychlost přenosu, jestliže je však záloha prováděna automaticky, je zbytečný. Přehledná statistika po ukončení zálohování jde zobrazit pomocí `--stats`.

Dalšími velice užitečnými přepínači jak u rsync tak u dalších použitých nástrojů jsou volby `--exclude` a `--include`. Díky nim lze přesně specifikovat, které adresáře nebo soubory chceme ze zálohy vyjmout nebo vynutit jejich zálohování. Oba tyto přepínače lze různě kombinovat.

Princip používání exclude a include je podrobně zobrazen v následujících ukázkách:

Následující příklad zálohuje domovský adresář uživatele student, ale vynechá ze zálohy adresář `/Videos/`.

```
rsync -avh --exclude=/Videos/ /home/student/ /media/backup/
```

Další ukázkou zálohuji opět domovský adresář uživatele student a vynechám vše z adresáře `/Videos/`, ale podadresář `/Videos/important/` bude do zálohy zahrnut.

```
rsync -avh --include=/Videos/important/ --exclude=/Videos/*
/home/student/ /media/backup/
```

Obdobně lze ze zálohy vynechat jen určité typy souborů. Další příklad vynechá vše s příponou .avi.

```
rsync -avh --exclude=*.avi /home/student/ /media/backup/
```

Jestliže je počet adresářů, které chceme ze zálohování vynechat nebo naopak zahrnout, malý, je postačující používat exclude a include jako ve výše popsaných příkladech. Pokud bychom ale měli tímto způsobem zapisovat větší množství adresářů (5 a více), stal by se kód nepřehledným. Z tohoto

důvodu zavádí zálohovací nástroje možnost specifikovat adresáře v textovém souboru, který je potom předán přepínači daného nástroje. U rsync k tomu slouží `--exclude-from=SOUBOR`.

Navržené řešení a popis vytvořeného skriptu

Rsync samotný nenabízí jednoduché řešení vytvoření inkrementální zálohy, ale je nutné v zálohovacím skriptu zajistit vše od základu jak vytváření adresářů, kam bude záloha ukládána, tak mazání starých záloh nebo případnou komprimaci. Ostatní nástroje si tyto věci buďto řeší samy, nebo k tomu mají určené přepínače. Toto je dáno tím, že rsync nevznikl původně jako nástroj pro zálohování dat.

Jedna z možností vytvoření inkrementálních záloh je využití pevných odkazů. Pevný odkaz je alternativní jméno pro soubor, který se nachází na jednom místě, ale můžeme k němu přistupovat pomocí různých jmen.

Navržený skript (řešení) využívá právě pevných odkazů. Při spouštění v jednodenním intervalu vytváří inkrementální zálohy, které se ukládají do samostatných adresářů a odpovídají sedmi dnům v týdnu (frekvence spouštění záloh je nastavitelná a zálohy starší než sedm dnů jsou mazány). Jednou týdně se provede archivace a komprese všech týdenních inkrementálních záloh pomocí programů tar a gzip, což zajistí dostupnost i již smazaných inkrementálních záloh.

Skript je rozdělen do dvou částí, přičemž první část se stará o vytváření inkrementálních záloh včetně úkonů s tím spojených (vytváření adresářů, mazání starých záloh atd.) Druhá část provádí archivaci a kompresi všech týdenních záloh. Celý skript včetně komentářů je k dispozici v příloze A. V další části jsou podrobněji popsány nejdůležitější úseky skriptu.

První část skriptu začíná zavedením proměnných, do kterých je uloženo aktuální a včerejší datum. Proměnná `TODAY` je využívána pro vytváření (pojmenovávání) nových adresářů pro inkrementální zálohy. Do proměnné `YESTERDAY` je uloženo den staré datum, to určuje cestu pro přepínač `--link-dest`, který vyhodnotí změny aktuálního stavu zdrojového adresáře vůči poslední záloze a zároveň vytvoří pevné odkazy na nezměněné soubory. Pro lepší orientaci jsou také cesty k zdrojovému a cílovému adresáři uloženy do proměnných. Poslední jsou přepínače, uloženy do `OPTIONS`. Kromě již dříve zmíněných přepínačů jsou zde také nové. Jejich popis je následující: `-z` zapne kompresi při přenosu, `--delete` odstraní soubory z cílového adresáře, pokud se již nenachází ve zdrojovém adresáři.

```
TODAY=`date -I`

YESTERDAY=`date -I -d "1 day ago"`

SOURCE="student@158.196.142.76:/home/student/"

TARGET="/media/backup/incremental/$TODAY"

LINK="/media/backup/incremental/$YESTERDAY"

OPTIONS="-ahze ssh --exclude=*.avi --include=/Pictures/Important/ --
exclude=/Pictures/* --exclude=/Downloads/ --delete --stats --link-
dest=$LINK"
```

Samotné spouštění zálohování provádí následující část skriptu. Tento řádek spustí rsync s parametry zadanými v OPTIONS a zálohuje z adresáře zadaného v SOURCE do adresáře TARGET.

```
rsync $OPTIONS $SOURCE $TARGET
```

Jak bylo zmíněno, rsync nemá přepínače, které by mazaly staré zálohy. Řešení mazání starých záloh je znázorněno na níže uvedené části skriptu. Díky tomu, že vytvořený skript chytře ukládá inkrementální zálohy do adresářů pojmenovaných podle data, lze zajistit mazání starých adresářů využitím této skutečnosti. Do proměnné `DAY7` se uloží sedm dní staré datum a smaže se adresář, který je takto pojmenovaný.

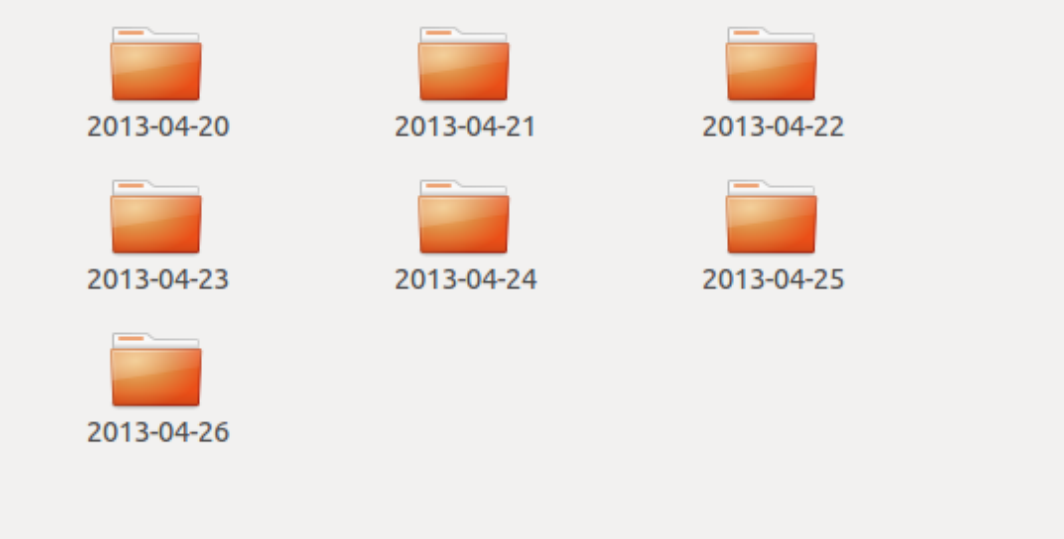
```
DAY7=`date -I -d "7 days ago"`

if [ -d /media/backup/incremental/$DAY7 ]

then

rm -R /media/backup/incremental/$DAY7

fi
```

Druhá část skriptu se stará o archivaci a kompresi všech týdenních záloh do jednoho archívu. Archivace a komprese je prováděna vždy v sobotu po ukončení zálohování, viz příloha A, ve které je celý skript. Následující ukázka z druhé části skriptu kontroluje počet archívů a maže je od nejstaršího v případě, že jejich počet přesáhne 7. Do proměnné `COUNT` je uložen údaj o počtu souborů v adresáři `/archive/`. Název nejstaršího souboru je uložen v proměnné `DEL_ARCH`. Podmínka testuje, zda je počet souborů větší než 7, jestliže je splněna smaže nejstarší soubor.

```
COUNT=`find /media/backup/archive/ -type f | wc -l`

DEL_ARCH=`ls /media/backup/archive/ -t | tail -n 1`

if [ $COUNT -gt 7 ]

then

rm -R /media/backup/archive/$DEL_ARCH

fi
```

Podoba výsledné zálohy

V adresáři, do kterého je záloha ukládána, jsou vytvořeny adresáře `/archive/` a `/incremental/`. Adresář `/incremental/` obsahuje maximálně sedm podadresářů, přičemž tyto podadresáře jsou pojmenovány po dnech, ve kterých byla záloha vytvořena, například 2013-04-21(viz Obrázek 3.1), a obsahují jednotlivé inkrementální rozdíly oproti předchozí záloze. Výhodou tohoto řešení je to, že i přestože se jedná o inkrementální zálohy, každý z těchto podadresářů obsahuje přesnou kopii stavu systému z daného dne, jakoby se jednalo o plnou zálohu. Toho je docíleno právě využitím pevných odkazů. Díky tomuto řešení je možné provádět rychlou obnovu dat prostým zkopírováním souborů.



Obrázek 3.1: Struktura inkrementálních záloh vytvořených skriptem pro rsync

V adresáři `/archive/` jsou uloženy komprimované archívy, které obsahují jednotlivé inkrementální rozdíly za celý týden. Počet těchto archívů, které budou udržovány, lze jednoduše nastavit, je ovšem nutné brát zřetel na to, aby tyto archívy nezabraly veškeré místo na pevném disku a jejich počet by měl být vhodně omezen. Případným lepším řešením by bylo tyto archívy přenášet opět na jiný vzdálený stroj.

Pro obnovení využijeme opět rsync a pouze přehodíme zdrojový adresář s cílovým adresářem. Obnovovat můžeme jednak jednotlivé soubory, tak i celé adresáře. V případě obnovování z tar archívů je potřeba nejprve archív rozbalit.

Souhrn vlastností a popis výhod a nevýhod navrženého řešení

Rsync je velice všestranný nástroj, ovšem pro vytvoření kvalitního zálohování je potřeba hlubších znalostí práce s tímto nástrojem, jelikož mnohé funkce, které nabídnou ostatní nástroje ve svém základu, u něj chybí. Svými vlastnostmi se dokáže velice dobře vyrovnat nástrojům, které jsou již přímo určeny k inkrementálnímu zálohování. Navíc lze ve velké míře ovlivnit způsob, jakým bude záloha vytvářena.

Pro určení změn v souborech si rsync vypočítává kontrolní součty ze souborů, které zálohuje a porovnává je s kontrolními součty souborů, které již zálohovány jsou. Rsync si ale tyto kontrolní součty, které už má vypočítanané, nikde neukládá a při další záloze je počítá znova. Jestliže by si vypočtené kontrolní součty uložil, při další záloze by se ušetřil čas potřebný pro výpočet kontrolních součtů. [11]

Do verze rsync 3.0.0 je seznam souborů, které mají být zálohovány, držen v paměti po celou dobu. Každý takovýto záznam potřebuje přibližně 100 bytu paměti. Od verze 3.0.0 je v paměti každý záznam udržován pouze tak dlouho, jak je to zapotřebí a snižuje tak využití paměti. Funguje, pouze když je na obou stranách verze 3.0.0 a vyšší. [18]

3.3 Řešení zálohování nástrojem rdiff-backup

Popis nástroje

Rdiff-backup je skript napsaný v pythonu a k zálohování využívá již popsaný rsync. Slouží k vytváření záloh jak na lokální disk, tak ke vzdálenému zálohování. Při vzdáleném zálohování je využívám protokol SSH. Převážně je rdiff-backup určen k vytváření inkrementálních záloh a jeho cílem je spojit ty nejlepší vlastnosti zrcadlení a inkrementální zálohy. Při zálohování zachovává

oprávnění, vlastnictví, pevné odkazy a další. Rsync doplňuje nejen o implementovanou podporu inkrementálního zálohování, ale i o další funkce jako je například snadné mazání starých záloh nebo uchovávání statistik o zálohách. Je-li vytvářena záloha na vzdálený stroj, je nutné, aby byl rdiff-backup nainstalován na lokálním i vzdáleném stroji, navíc je háklivý na různé verze, proto je potřeba, aby byly verze na obou strojích stejné. [16]

Základní práce

Základní práce s rdiff-backup je obdobná jako u rsync a zobrazuje ji následující ukázka.

```
rdiff-backup -v3 /home/student/ student@IP::/media/backup/
```

Jediným rozdílem v případě základního použití jak pro místní tak i pro vzdálenou zálohu jsou dvě dvojtečky u cílového adresáře místo jedné, jako tomu bylo u rsync. Nutná je také přítomnost rdiff-backup na vzdáleném stroji a to nejlépe ve stejné verzi.

Výše zmíněný příkaz při prvním spuštění vytvoří plnou zálohu, ale při opakujícím se spuštění již vytváří inkrementální zálohy. Na rozdíl od rsync u něj nemusí být nastaveno ručně, které adresáře se mají porovnávat a vše se řeší automaticky.

Nespornou výhodou rdiff-backup oproti rsync je snadné mazání starých záloh pomocí k tomu určených přepínačů, díky čemuž odpadá značná část práce. Možnosti jak smazat staré zálohy jsou dvě a u obou se využije přepínač `--remove-older-than`. První možností je mazat staré zálohy podle stáří. Tehdy se přepínač `--remove-older-than` doplní například o `1M`, což značí „starší než“ jeden měsíc.

K dispozici jsou následující parametry: s, m, h, D, W, M, nebo Y a určují sekundy, minuty, hodiny, dny, týdny, měsíce nebo roky. Lze použít i jejich kombinace, jako například 4W3D5h, což smaže zálohy starší než 4 týdny, 3 dny a 5 hodin. Praktickou ukázku zobrazuje následující příkaz.

```
rdiff-backup --remove-older-than 1M --force
student@IP::/media/backup/
```

Druhou možností je od verze 0.13.1 využití mazání s příponou B, která značí počet záloh, které budou zachovány.

Navržené řešení a popis vytvořeného skriptu

Použití zálohovacího nástroje rdiff-backup je oproti rsync o poznání jednodušší. Jelikož se jedná o nástroj určený přímo pro vytváření inkrementálních záloh, není nutné zajišťovat vytváření adresářů, kam by se ukládaly jednotlivé inkrementální zálohy. Všechny úkony spojené s vytvářením adresářů si řeší rdiff-backup sám. Výsledný skript je díky této skutečnosti mnohem kratší.

Pro zálohování dat pomocí tohoto nástroje bylo opět zvoleno inkrementální zálohování. Skript je možné spouštěn v pravidelných intervalech pomocí cronu. Při prvním spuštění je vytvořena plná záloha a při dalším spuštění jsou již vytvářeny inkrementální zálohy. Oproti předchozímu řešení rdiff-backup udržuje pouze jednu nekomprimovanou kopii dat a samotné inkrementální rozdíly si ukládá do speciálního adresáře rdiff-backup-data v komprimované podobě.

Celý skript je přiložen, jako příloha B. Skript pro rdiff-backup začíná stejně jako předchozí pro rsync, tedy uložením potřebných údajů do proměnných viz příloha B. Tentokrát je ovšem počet proměnných menší a vystačí si s `SOURCE`, `TARGET` a `OPTIONS`.

Dále je vytvořena proměnná `OPTIONS_RM` ve které jsou pokyny pro mazání starých záloh a následuje spuštění zálohování a nakonec spuštění mazání starých záloh.

```
OPTIONS_RM="--remove-older-than 1M --force"

rdiff-backup $OPTIONS $SOURCE $TARGET

rdiff-backup $OPTIONS_RM $TARGET
```

Aby mohly být staré zálohy smazány, musí se rdiff-backup spustit zvlášť pro zálohování a následně pro mazání, jelikož přepínač `--remove-older-than` nejde použít společně s přepínači v proměnné `OPTIONS`.

Z popisu skriptu je zřejmé, o kolik jednodušší je vytvoření skriptu pro rdiff-backup oproti rsync.

Podoba výsledné zálohy

V cílovém adresáři, jsou vždy k dispozici přesné kopie dat v nekomprimované podobě, která odpovídají stavu systému po poslední záloze a slouží k rychlému obnovení prostým zkopírováním. Pro obnovení starších záloh je nutné použít rdiff-backup. Navíc je vytvořen adresář rdiff-backup-data, do kterého se ukládají statistiky a metadata. V tomto adresáři je také podadresář increments, do kterého se ukládají při dalším spuštění zálohy inkrementální změny.

V případě obnovování souborů starších než ty, které jsou v nekomprimované podobě je potřeba využít-rdiff backup s přepínačem –r. Pomocí přípon, s, m, h, D, W, M, nebo Y určujeme bod v čase, ke kterému bude záloha obnovena.

```
rdiff-backup –r 2M /media/backup /home/student/restore
```

Výše zmíněny příkaz provede obnovení 2 měsíce staré zálohy. S velkou pravděpodobností neexistuje záloha, která by byla přesně 2 měsíce stará. V těchto případech se obnovuje od nejbližší předchozí zálohy. Obnovit lze i konkrétní soubor jak ukazuje následující ukázka.

```
rdiff-backup –r 2M /media/bakalářka.pdf /home/student/bakalářka.pdf
```

Souhrn vlastností a popis výhod a nevýhod navrženého řešení

Na rozdíl od řešení zálohování v podobě rsync nenabídne rdiff-backup sedm posledních plných záloh v otevřené podobě, ale pouze jednu. Jeho výhodou je, že i přestože se jedná o nástroj určený pro terminál, je snadno ovladatelný a nabídne uživateli množství užitečných funkcí. Podobně jako rsync si kontrolní součty nikam neukládá a při každé následující záloze je vypočítává znova. Navíc je citlivý na různé verze rdiff-backup při vzdáleném zálohování a je vhodné mít na obou strojích stejné verze. Jistou nevýhodou může být i ta skutečnost, že inkrementální zálohy mohou tvořit dlouhou řadu na sebe navazujících záloh, protože je vytvořena pouze jedna plná záloha.

3.4 Řešení zálohování nástrojem duplicity

Popis nástroje

Duplicity je jeden z dalších nástrojů, které využívají rsync algoritmus. Duplicity zálohuje soubory do zvoleného adresáře a ihned vytváří tar archívy, není tedy k dispozici žádná otevřená podoba zálohy a k obnově je vždy třeba duplicity. Zálohovací nástroj duplicity přináší další řadu funkcí, tou hlavní je šifrování záloh pomocí GnuPG, díky čemuž můžeme zálohovat i na servery, které nejsou pod naší kontrolou. Odpadá také nutnost mít duplicity nainstalován na obou strojích v případě, že zálohujeme vzdáleně. Jednoduše jdou také vytvářet plné zálohy na „povel“.

Narozdíl od předchozích nástrojů má duplicity jednu další významnou výhodu. Tou výhodou je to, že si během vytváření zálohy zaznamenává vypočtené kontrolní součty. Když je poté vytvářená inkrementální záloha, není již nutné přečíst celou předchozí zálohu, aby mohly být určeny změny, ale stačí pouze porovnat již dříve spočtené kontrolní součty. Také umožňuje zachovat práva a vlastnictví i na médiích, která to neumožňují. [11]

Základní práce

Práce s duplicity je velice podobná práci s rdiff-backup. Hlavní rozdíl je, že v případě duplicity se cílový adresář, zadává ve formátu URL a ve výchozím nastavení duplicity šifruje.

Místní záloha se tedy provede následujícím příkazem:

```
duplicity --no-encryption /home/co file://media/backup/
```

V tomto případě `--no-encryption` značí, že je šifrování zakázáno a duplicity nebude vyžadovat heslo (myšleno heslo pro šifrování, heslo pro připojení na vzdálený stroj vyžadováno bude). To by stačilo pro vytváření místní zálohy, jestliže ale chceme zálohovat na vzdálený stroj, který nemáme zcela pod kontrolou, nebo k němu mají přístup neoprávněné osoby, je vhodné využít výhody duplicity a použít šifrování.

Vzdálená záloha se provede obdobně. Pro přenos zálohy lze využít bezpečné kopírování SCP,FTP, rsync a další.

```
duplicity /home/student/ scp://uzivatel@media/backup/
```

Stačí tedy vynechat parametr `--no-encryption` a záloha bude šifrována. Po spuštění příkazu bude vyžadováno heslo pro zašifrování. Jestliže chceme celý proces automatizovat, je psaní hesla po spuštění skriptu nežádoucí, jak tento problém obejít názorně ukážu v popisu navrženého skriptu.

V případě použití základního šifrování u duplicity se jedná o jednoduchou symetrickou šifru. Ta ovšem není zcela bezpečná. Nicméně nám duplicity poskytuje řešení i na tento problém a to je použití asymetrických klíčů. ID klíče pak lze předat jako parametr pro `--encrypt-key`.

Navržené řešení a popis vytvořeného skriptu

Pro zálohování nástrojem duplicity bylo využito jeho hlavní výhody oproti předchozím nástrojům a záloha byla šifrována symetrickou šifrou. Navržené řešení vytváří nejprve plnou zálohu a v následujících dnech jsou vytvářeny inkrementální zálohy. Plná záloha je vytvořena znova, pokud je ta předchozí starší než sedm dnů. Díku tomu je zajištěna vetší bezpečnost protože nemáme vytvořenou dlouhou řadu na sebe navazujících inkrementálních záloh jako u rdiff-backup. Udržovány, jsou 3 staré plné zálohy a na ně navazující inkrementální zálohy. Počet plných záloh je snadno nastavitelný.

Okomentovaný skript je k dispozici v příloze C, další část popíše jednotlivé části skriptu a vysvětlí jejich funkci.

Skript začíná již klasickým zadáním zdrojového adresáře a cílového adresáře. Pro samotný přenos zálohy na vzdálený stroj byl použit rsync. Při použití SCP docházelo k problémům, které jsou zmíněny v kapitole 4. této bakalářské práce. V případě přepínačů `OPTIONS` se objevuje nový přepínač `full-if-older-than`. Díky tomuto přepínači můžeme snadno určit, v jakém intervalu bude prováděna kompletní plná záloha.

```
SOURCE="/home/student/"

TARGET="rsync://student@158.196.142.77//media/backup/"

OPTIONS="--exclude /home/student/**.avi --include
/home/student/Pictures/Important --exclude /home/student/Pictures --
exclude /home/student/Downloads --full-if-older-than 7D"
```

Další důležitou částí v případě, že chceme, aby byla záloha šifrována a prováděna automaticky, je uložení hesla do proměnné `PASSPHRASE`. Poté již můžeme duplicity spustit a zadané heslo bude použito pro zašifrování.

```
export PASSPHRASE='dlouhe_tajne_heslo'

duplicity $OPTIONS $SOURCE $TARGET
```

Duplicity obdobně jako rdiff-backup nemůže mazat staré zálohy během zálohování a je nutné ho spustit znovu. `Remove-all-but-n-full` „parametr“ určuje počet plných záloh a na ně navazujících záloh které budou zachovány.

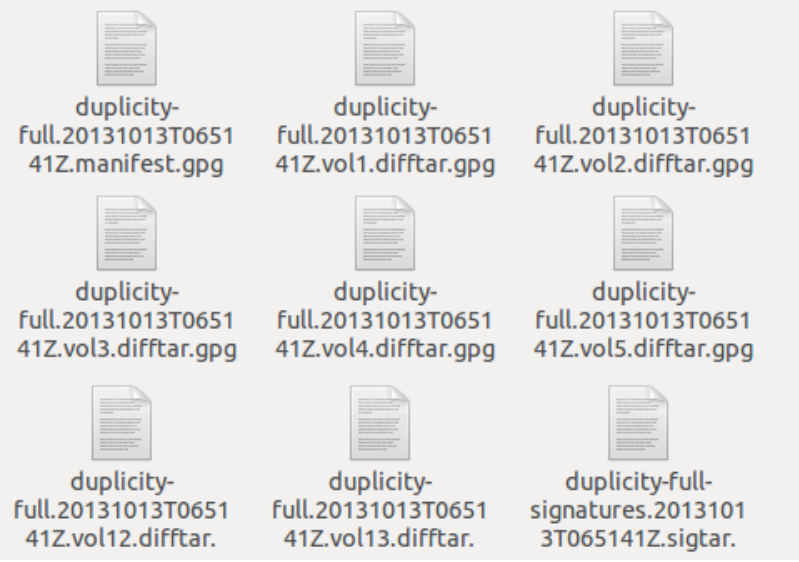
```
OPTIONS_RM="remove-all-but-n-full 3 --force"

duplicity $OPTIONS_RM $TARGET
```

Podoba výsledné zálohy

V cílovém adresáři je vytvořeno několik souborů. Jestliže není záloha šifrována, jedná se o difftar.gz, manifest (textový soubor) a sigtar.gz. A v případě použití šifrování jsou to soubory difftar.gpg, manifest.gpg a o sigtar.gpg.

Samotná data jsou v souborech difftar.gz což jsou tar archívy. Jestliže je potřeba nějaký soubor obnovit, musí být přečten celý archív, což by mohlo trvat dlouhou dobu, proto duplicity omezuje velikost tar archívu na 26 MB a poté vytváří další. Samotná informace o tom, ve kterém archívu je žádaný soubor uložen, je uložena v textovém souboru manifest. Aby bylo možné určit, které soubory byly změněny nebo smazány, je vytvořen ještě sigtar, který obsahuje kontrolní součty. Kontrolní součty i s manifestem nejsou uloženy pouze na vzdáleném stroji, ale i na stroji, který zálohujeme typicky v `~/.cache/duplicity/`.[12]

Na obrázku 3.2 je zobrazeno, jak vypadá výsledná záloha. Je patrné vidět, že záloha je šifrována.



Obrázek 3.2: Záloha vytvořena skriptem pro duplicity (výřez)

Pro obnovení dat z poslední vytvořené zálohy stačí přehodit pořadí zdrojového a cílového adresáře. V případě použití symetrické šifry bude vyžádáno heslo. Jestliže by byla použita asymetrická šifra je potřeba ji specifikovat pomocí --encrypt-key.

Obdobně jako u rdiff-backup lze obnovit zálohu od konkrétního data. K tomu slouží u duplicity přepínač `-t` a jeho parametr. Obnovení z 20 dnů staré zálohy provede následující příkaz.

```
duplicity -t 20D rsync://student@IP/backup/ /home/student/restore/
```

K obnovení konkrétního souboru či adresáře slouží volba --file-to-restore.

Souhrn vlastností a popis výhod a nevýhod navrženého řešení

Z testovaných nástrojů je duplicity společně s rsync nejzajímavější. Rsync vynikl ve své škálovatelnosti a v širokých možnostech, jak se dá se zálohou pracovat, byť je třeba hlubších znalostí práce s OS Linux. Duplicity oproti tomu nabízí možnosti jak kvalitně a hlavně přehledně zálohovat, přičemž velice zjednodušuje tuto práci.

Velkou výhodou je bezesporu také to, že si ukládá kontrolní součty již vytvořených záloh. Jako příklad dalších užitečných funkcí lze zmínit mazání starých záloh podle stáří pomocí `remove-older-than 60D`. Dojde k smazání plných záloh a na ně navazujících inkrementálních záloh starších než 60 dní. Jednoduše můžeme také vyčistit úložiště od nepotřebných nebo poškozených záloh díky `cleanup`.

3.5 Řešení zálohování nástrojem hdup2

Popis nástroje

Hdup2 je již vytvořený zálohovací skript, jehož autorem je Miek Gieben, který se v současnosti podílí na vývoji zálohovacího nástroje rdup. Hdup2 je nástroj určený pro inkrementální zálohování jak lokální, tak i vzdálené [17]. Vývoj tohoto nástroje byl již ukončen. Přestože se jedná o nástroj pro terminál, jeho používání je velice snadné a spočívá v pouhé úpravě konfiguračního souboru. Pro přenos záloh na vzdálený stroj používá protokol SCP.

Mezi další možnosti hdup2 patří komprese pomocí bzip, gzip, lzop, ale lze použít i metodu bez komprese. Zároveň zajišťuje možné rozdělení archívů podle velikosti. To může byt výhodné tam, kde chceme výslednou zálohu například vypalovat na CD/DVD. V neposlední řadě může hdup2 zálohu také šifrovat a to buď pomocí mcrypt nebo GPG. [18]

Nevýhodou tohoto jinak povedeného skriptu je absence možnosti include nebo přehlednější možnosti mazání starých záloh.

Hdup2 vytváří zálohy v těchto režimech:

Měsíční záloha je plná záloha všech vybraných dat.

Týdenní záloha je inkrementální záloha vztažena k poslední měsíční záloze.

Denní záloha je vztažena k posední týdenní záloze.

Z těchto režimů, ve kterých hdup2 pořizuje zálohy, vyplývá, že se jedná také o diferenciální zálohy. Jelikož týdenní záloha se vztahuje k poslední měsíční záloze, tak jsou při každém vytvoření týdenní zálohy zálohována i ta data, která obsahuje předchozí týdenní záloha a přitom se nezměnila. Stejně tak jsou denní zálohy tvořeny od poslední týdenní zálohy.

Základní práce

Veškerá nastavení se provádí v konfiguračním souboru, který je standardně umístěn v `/etc/hdup/hdup.conf`.

Pro vytvoření nejjednodušší zálohy stačí upravit následující řádky.

```
archive dir = /media/zaloha/

user = student
```

V první řadě je tedy vybrán cílový adresář, a další řádek nastaví vlastníka vytvořené zálohy.

Nakonec v sekci `#my own host` změníme název profilu a vybereme zdrojový adresář.

```
[student]

dir = /home/student/
```

Profilů může být vytvořeno více a samotná záloha se poté spouští zadáním názvu profilu.

Spuštění zálohy provedeme takto: `hdup monthly student`. Spustí se měsíční plná záloha podle parametrů nastavených v profilu student. Týdenní a denní záloha se provede pouze změnou monthly na weekly nebo daily. Samotné názvy monthly, weekly a daily neurčují intervaly spouštění, ale pouze to, jaký typ zálohy se vytvoří. O automatizaci spouštění se musí postarat cron.

Práce se zálohováním na vzdálený stroj je odlišná, než u předchozích nástrojů. Za předpokladu, že máme na lokálním stroji již vytvořen profil student popsány výše, pak na vzdáleném stroji provedeme úpravy v konfiguračním souboru takto:

Změníme dva řádky. Tím prvním je známé nastavení cílového adresáře pro uložení zálohy a další povolí vzdálené zálohování.

```
archive dir = /zaloha/

allow remote = yes
```

A rovněž vytvoříme profil student, který bude úplně stejný jako na lokálním stroji, který zálohujeme.

```
[student]

dir = /home/student/
```

Tím je vše nastaveno a stačí spustit hdup2 na lokálním stroji. Vzdálený stroj se opět zadává trochu rozdílně oproti předchozím nástrojům, adresář již není nutné zadávat.

```
hdup monthly my-comp @student@192.168.80.197
```

Navržené řešení a popis vytvořeného skriptu

V případě navržení řešení pro hdup2 se nejedná ani tak o vytvoření skriptu jakožto o jeho nastavení. Nastavení hdup2 spočívalo v úpravě skriptu na lokálním a vzdáleném stroji. Byl nastaven adresář, kam se bude zálohovat.

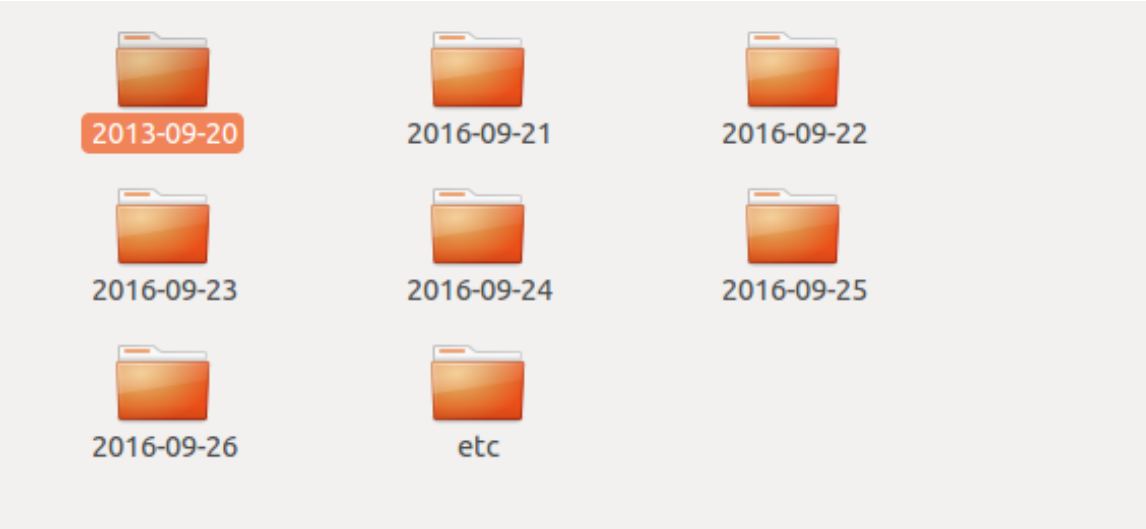
Dále byl vytvořen profil student, který vytváří zálohu domovského adresáře student, přičemž ze zálohy vyjme podadresář Downloads. Na rozdíl od předchozích nástrojů nemá hdup2 možnost include, rovněž nejde vyloučit ze zálohy pouze určité typy souborů. Konfigurační soubor je přiložen jako příloha L. O pravidelné spouštění správného typu zálohy by se staral démon cron. Možné nastavení by mohlo vypadat jako ukázky níže, ty spustí hdup2 nejprve jako měsíční zálohu vždy první den v měsíci, pak týdenní zálohu každých sedm dní a nakonec denní zálohy každý den.

```
00 00 1 * * root /usr/local/sbin/hdup -q monthly student
@student@158.196.142.77

00 05 1,7,14,21,28 * * root /usr/local/sbin/hdup -q weekly student
@student@158.196.142.77

00 08 1-31 * * root /usr/local/sbin/hdup -q daily student
@student@158.196.142.77
```

Další drobnou nevýhodou hdup2 je to, že sám neumí mazat staré zálohy pomocí přepínačů, jako to uměly rdiff-backup a duplicity, proto je potřeba si pomoct dalším skriptem. Naneštěstí ukázka toho, jak by se mohly mazat staré soubory, je v instalačním balíčku hdup2, nebo lze využít část skriptu pro rsync.

Podoba výsledné zálohy

V cílovém adresáři, kam byla záloha uložena, dojde k vytvoření adresáře pojmenováno jako zálohovací profil - v našem případě „student“. V tomto adresáři se nachází adresář /etc/, do kterého jsou ve formě textových souborů ukládány záznamy o provedených zálohách a inkrementální změny. V adresáři /etc/ se rovněž vytváří adresáře pojmenované podle data obdobně, jako tomu bylo u rsync. Do těchto adresářů jsou ukládány komprimované archívy se zálohami měsíčního, týdenního nebo denního typu. Obnovení ze zálohy se provede příkazem: hdup restore student 2008-01-31 /restore/



Obrázek 3.3: Struktura záloh vytvořených skriptem hdup2

Souhrn vlastnosti a popis výhod a nevýhod navrženého řešení

Ovšem absence možnosti include a tím pádem absence možnosti zálohovat pouze určité typy souborů a zároveň možnost vynechávat pouze určité typy souborů je vzhledem k ostatním nástrojům podstatná. Navíc nedosahuje takové podpory linuxových uživatelů, jako tomu bylo u všech předchozích nástrojů. Vzhledem k těmto okolnostem je lepší volnou výběr některého z předchozích nástrojů, zvláště duplicity.

3.6 Porovnání řešení

V průběhu této kapitoly, jsem postupně popisoval výhody a nevýhody jednotlivých řešení a nástrojů. V této podkapitole bych chtěl navržené řešení porovnat mezi sebou. Mezi nejzajímavější testované nástroje patřil duplicity a rsync. V porovnávání se zaměřím hlavně na porovnání duplicity s ostatními nástroji.

Duplicity ve své funkcionalitě oproti rdiff-backup a hdup2 velice vyniká. Jeho možnost šifrovat zálohy nebo to, že nemusí být nainstalován i na vzdáleném stroji, na který je záloha ukládána z něj činí velice kvalitní a všestranný zálohovací nástroj. Struktura záloh, kterou navržené řešení pomoci nástroje duplicity vytváří, je rovněž bezpečnější, jelikož vytváří novou plnou zálohu, pokud ta předchozí je starší než sedm dnů. Stará plná záloha je společně s navazujícími inkrementálními zálohami ponechána. Řešení pomoci rdiff-backup vytváří dlouhou řadu na sebe navazujících inkrementálních záloh, které nejsou doplněny o více plných záloh. Obdobně je tomu u hdup2, kde je vytvořena měsíční záloha, následuje týdenní záloha a na ni navazující denní zálohy.

V případě porovnání rdiff-backup s duplicity a ze skutečností zjištěných při testování, nenabízí rdiff-backup krom nekomprimované zrcadlové kopie dat žádnou podstatnou funkci navíc. Vzhledem k úsilí, které musí být vynaloženo pro vytvoření kvalitního zálohovacího procesu, je vhodnější nasadit duplicity ihned i přesto, že některé z jeho funkcí nemusí být v daném okamžiku využity (například šifrování), než nasadit rdiff-backup. Protože v případě, že by nastala potřeba šifrovat zálohy, muselo by se složitě přecházet na jiný nástroj.

Hdup2 sice stejně jako duplicity nabídne možnost šifrovat zálohy, ale nedokáže ze zálohy vynechávat určité typy souborů. Také u něj chybí možnost include a vzhledem k tomu, že duplicity obě tyto možnosti nabízí, je lepší volbou.

Samotnou kapitolu tvoří rsync a řešení navržené pomocí tohoto nástroje. V nekomprimované podobě je uchováváno hned sedm inkrementálních záloh, přičemž tyto zálohy zobrazují přesný stav zdrojového adresáře v dobu, kdy byla záloha vytvořena jakoby, se jednalo o plnou zálohu. Podstatným faktem je to, že zabírají prostor pouze jako inkrementální záloha. Těchto sedm adresářů je pravidelně archivováno a pro další zvýšení bezpečnosti by mohly být ukládány opět na jiný disk, než jsou uloženy nekomprimované podoby zálohy. Slabinou tohoto řešení je to že na pevném disku nebo USB flash disku, kam je záloha ukládána musí být stejný souborový systém a musí podporovat pevné odkazy.

Při porovnání skriptů vytvořených pro duplicity a rsync jsou rozdíly jasně patrné. V případě duplicity, který je určen přímo k zálohování, si skript vystačí několika přepínači, díky kterým lze snadno specifikovat podobu výsledné zálohy. U rsync je potřeba hlubších znalostí práce s OS Linux, jelikož je skript tvořen od základu (vytváření adresářů, mazání starých záloh atd.). Konkrétní navržená řešení se od sebe také značně liší. Výhody a nevýhody obou řešení již byla zmíněna. V případě rsync by bylo možné skript dále rozšiřovat a doplnit ho třeba o šifrování vytvářených archívů.

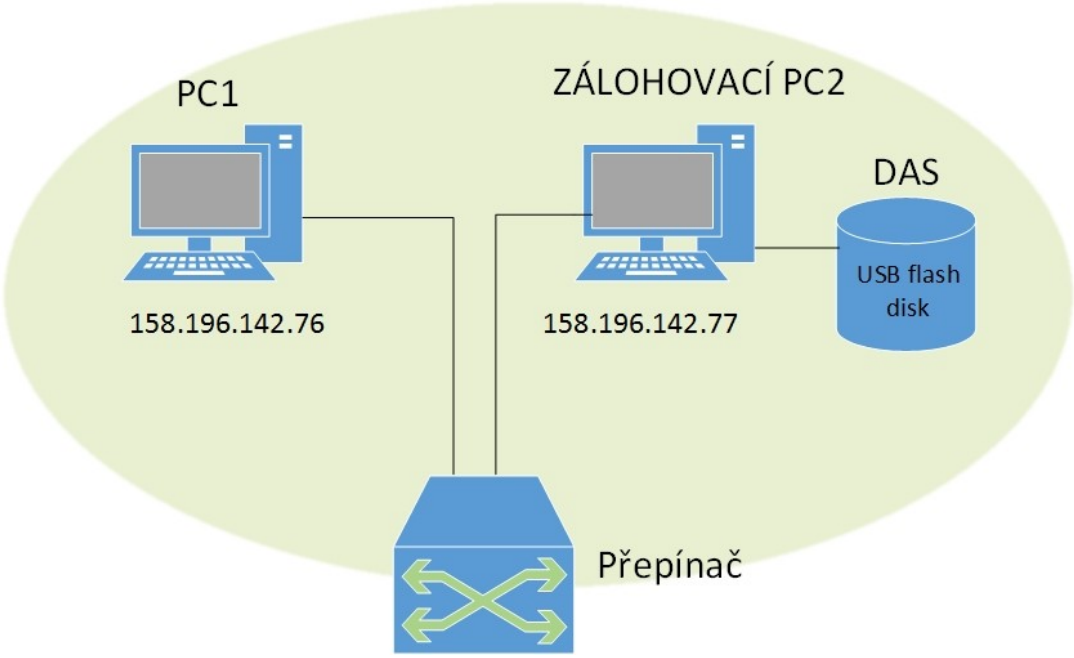
4 Ověření navržených řešení v laboratorních podmínkách

Poslední kapitola bakalářské práce popisuje průběh laboratorního testování. Všechna navržená řešení jsem ověřil v laboratorních podmínkách. Pro otestování vytvořených skriptů, byla prováděna záloha jednoho PC na jiný PC v téže síti LAN. Vytvořená záloha se ukládala na datové úložiště typu DAS a konkrétně se jednalo o USB flash disk naformátovaný na ext4. Na PC v laboratoři je nainstalován OS Ubuntu verze 12.04 LTS 32-bit.

Podstatou této kapitoly je tedy ukázat, že navržená správně vytváří inkrementální zálohy zdrojového adresáře. Jako zdrojový adresář byl vybrán domovský adresář uživatele Ubuntu. Dále budou v této kapitole popsány statistiky po ukončení zálohy. V případě inkrementální zálohy by mělo být ze všech statistik patrné, že se provedla záloha pouze změněných nebo nových dat.

4.1 Schéma zapojení

Ověření navržených řešení probíhalo na zapojení podle obrázku 4.1. PC1 obsahoval data, která byla zálohována a ty se pak ukládala na USB flash disk připojený k ZÁLOHOVACÍ PC2. PC1, který se zálohoval, měl nastavenou IP adresu 158.196.142.76 a ZÁLOHOVACÍ PC2 disponoval IP adresou 158.196.142.77, viz obrázek 4.1.



Obrázek 4.1: Schéma zapojení

4.2 Konfigurace RSA klíčů

I přestože při laboratorním měření byly skripty spouštěny ručně, bylo vhodné nastavit SSH přístup na vzdálený stroj tak, aby připojení proběhlo bez vyžádání hesla. V opačném případě by si každé opětovné spuštění skriptu vyžádalo ruční zadání hesla. Ruční spouštění skriptů při testování jsem zvolil z důvodu vyšší přehlednosti při testování a snadnější realizaci. Generování RSA klíčů a jejich zprovoznění je popsáno v příloze D. Při praktickém nasazení navržených řešení by se o spouštění skriptů staral démon cron, aby byla splněna podmínka automatického zálohování.

Popis práce s démonem cron je popsán v části 3.1 této bakalářské práce. Pro automatické spouštění navržených skriptů by tedy stačilo využít poznatků z 3.1.

4.3 Popis průběhu laboratorního testování

Pro otestování navržených skriptů jsem zvolil vytvoření zálohy domovského adresáře uživatele „student“ z PC1 na ZÁLOHOVACÍ PC2. Cesta k tomuto adresáři je `/home/student/`. V tomto domovském adresáři byla adresářová struktura doplněna o vhodné typy souborů pro demonstraci zálohování.

Typický domovský adresář uživatele Ubuntu obsahuje několik dalších adresářů, jako jsou: `/Desktop/`, `/Downloads/`, `/Pictures/`, `/Videos/` nebo `/Documents/` a velké množství skrytých souborů a adresářů (například .mozilla, ve kterém jsou uživatelské nastavení prohlížeče Firefox). Do adresáře `/Videos/` jsem přidal soubor s příponou .avi o velikosti 65 MB. Adresář `/Downloads/` měl velikost 32 MB a celková velikost adresáře `/student/` byla 277 MB.

Skripty, které prováděly zálohu byly nastaveny tak, aby nezálohovaly soubory s příponami .avi, dále vynechaly celý adresář `/Downloads/` a z adresáře `/Pictures/` zálohovaly pouze podadresář `/Important/` (více o nastavení jednotlivých skriptů je zmíněno v předchozí kapitole 3., která popisuje navržené skripty).

Při prvním spuštění všech skriptů se nejprve automaticky provedla plná záloha, jelikož neexistovala předchozí záloha, od které by se mohly tvořit navazující inkrementální zálohy. Tato první záloha obsahovala vše, co bylo do zálohy zahrnuto. Celkem se jednalo o 180 MB dat, což odpovídá velikosti adresáře `/student/` bez adresáře `/Downloads/`, adresáře `/Pictures/` mimo podadresář `/Importatnt/` a .avi souboru, které nemají být zálohovány. Velikost přenesených dat nemusí být vždy přesně 180 MB, jelikož některé nástroje si vytvářejí pomocné soubory.

Před opětovným spuštěním skriptu jsem provedl změnu data příkazem **DATE** jehož názorná ukázka použití vypadá takto.

```
DATE 0406162013
```

První dvojčíslí označuje měsíc, druhé dvojčíslí den, třetí hodinu a poslední čtyřčíslí rok. Měnit den před opětovným spuštěním skriptů je důležité zvlášť u skriptu rsync, protože jeho správná funkce je založena na vytváření a mazání adresářů podle dnů. Jestliže by nebyl pro navržený skript dodržen jednodenní interval bez patřičných úprav skriptu, došlo by k chybám ve struktuře záloh (například vytváření plných záloh, přestože má být vytvořena inkrementální).

Poté co již byla vytvořena plná záloha právě testovaným skriptem, jsem do adresáře `/Documents/` umístil nový soubor o velikosti 46 MB (s příponou .exe). Skript spuštěný po přidání tohoto nového souboru zálohuje už pouze tento nově vzniklý soubor a ne znova celý adresář `/student/`, vytváří tedy inkrementální zálohu.

Verze použitých nástrojů:

- Rsync 3.0.9
- Rdiff-backup 1.2.8
- Duplicity 0.6.18
- Hdup2 2.0.14

4.4 Popis statistik vytvořených záloh

RSYNC

Skript pro RSYNC byl jako jediný spouštěn ze zálohovacího PC2 a přenášel zálohu z PC1 na ZÁLOHOVACÍ PC2.

Níže je uveden výpis celé statistiky po dokončení první plné zálohy. Nejdůležitější jsou údaje o počtu přenesených souboru (Number of files transferred), celková velikost dat, která mají být přenesena (Total file size) a velikost dat, která byla přenesena (Total transferred file size). Údaj total bytes received zobrazuje, kolik dat bylo reálně přeneseno přes síť. Právě tento údaj je ovlivněn kompresí –z. Bez zapnuté komprese by se jeho velikost shodovala s total transferred file size. Údaj total size ukazuje celkovou velikost cílového adresáře.

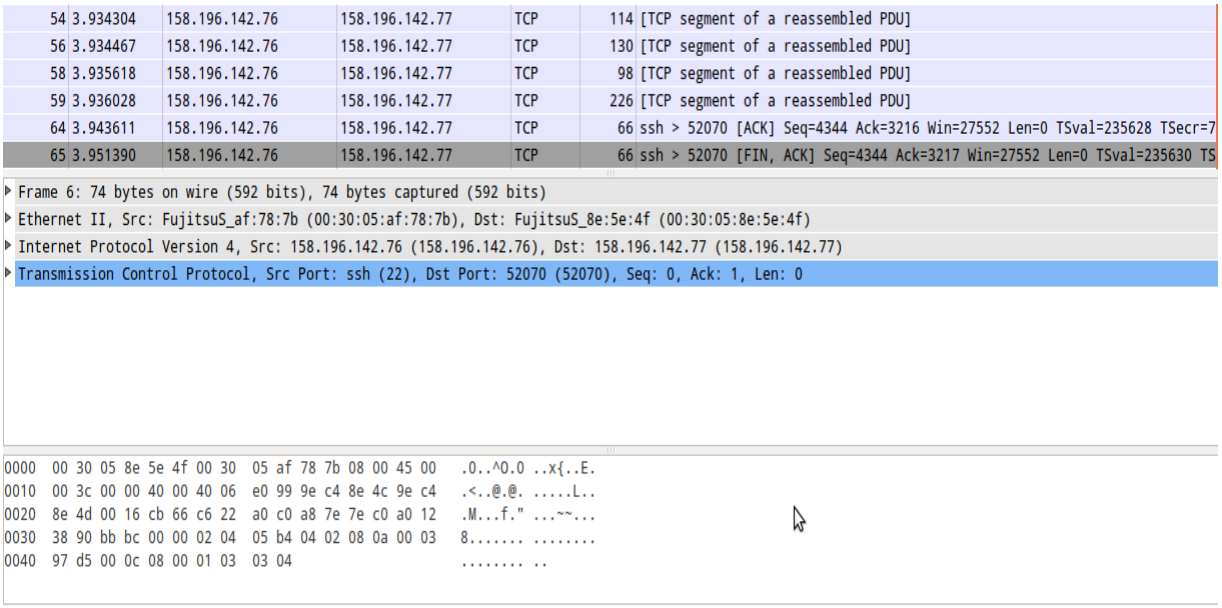
```
Number of files: 1478

Number of files transferred: 1116

Total file size: 181.95M bytes

Total transferred file size: 181.95M bytes

Literal data: 181.95M bytes

Matched data: 0 bytes

File list size: 37.73K

File list generation time: 0.004 seconds

File list transfer time: 0.000 seconds

Total bytes sent: 22.75K

Total bytes received: 86.68M


sent 22.75K bytes  received 86.68M bytes  854.21K bytes/sec

total size is 181.95M  speedup is 2.21
```

Z následující ukázky části výpisu po ukončení inkrementální zálohy je již patrné, že v adresáři, který byl zálohován, došlo ke změnám. Zvětšila se celková velikost zdrojového adresáře (o nový soubor 46 MB). Dále je z výpisu vidět, že byly přeneseny pouze změny od poslední zálohy, tedy celková velikost dat, přenesených přes síť byla 47,96 MB, kdežto u předchozí plné zálohy to bylo 86,68 MB. Dále je také z výpisu patrná celková velikost cílového adresáře, do kterého byla záloha uložena (total size). Celý výpis je k dispozici v příloze E.

```
Total file size: 230.07M bytes
Total transferred file size: 49.42M bytes
Total bytes received: 47.96M

sent 8.88K bytes  received 47.96M bytes  1.28M bytes/sec
total size is 230.07M  speedup is 5.00
```

Ze statistik lze tedy vyvodit, že byla vytvořena inkrementální záloha. Jestliže, by byl některý soubor z adresáře `/student/` smazán a zároveň by nebyl vytvořen žádný nový, pak by nebyla přenesena žádná data a soubor bude přístupný už pouze ve starší záloze případně v komprimované podobě. V nekomprimované podobě je dostupný, dokud bude existovat alespoň jeden pevný odkaz na něj.

Na obrázku 4.2 je zachycen přenos zálohovaných dat skriptem rsync přes síť programem wireshark. na obrázku je vidět přenos zálohy a také ukončení šifrované komunikace. Také je patrné, že přenos zálohy probíhá šifrovaně. Obdobný průběh mají i ostatní nástroje, proto již není dále zmiňován.



Obrázek 4.2: Přenos zálohy nástrojem rsync

RDIFF-BACKUP

Skript vytvořený pro rdiff-backup byl spouštěn z PC1 a přenášel zálohu na ZÁLOHOVACÍ PC2.

Následující část výpisu ukazuje nejdůležitější části ze statistiky po ukončení první zálohy. Opět můžeme vyčíst velikost zálohovaného adresáře (SourceFileSize), velikost nových souborů, které ještě nebyly zálohovány (NewFileSize) a pak také výslednou změnu velikosti cílového adresáře, kam byla záloha uložena (TotalDestinationSizeChange). Celý výpis je v příloze F.

```
SourceFileSize 191763792 (183 MB)

NewFileSize 191763792 (183 MB)

TotalDestinationSizeChange 191763792 (183 MB)
```

Z výpisu statistiky po ukončení inkrementální zálohy lze vidět změnu ve velikosti zdrojového adresáře `/student/` (předchozí stav 183 MB, nový stav 229 MB), velikost shodných dat zdrojového adresáře s předchozí zálohou (MirrorFileSize). V neposlední řadě, je také uvedena velikost nových souborů (NewFileSize), viz následující ukázka a velikost nových dat v cílovém adresáři. Celá statistika je přiložena jako příloha G.

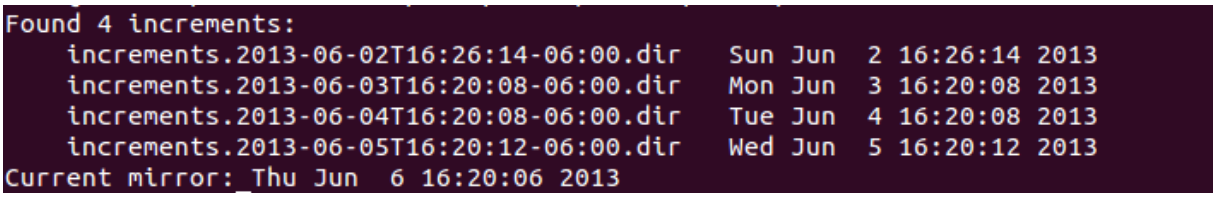
```
SourceFileSize 240019180 (229 MB)

MirrorFileSize 191763792 (183 MB)

NewFileSize 48111507 (45.9 MB)

TotalDestinationSizeChange 51502054 (49.1 MB)
```

Ze statistik vyplývá, že byla provedena inkrementální záloha. Komprese při přenosu je v případě rdiff-backup zaplá automaticky.

Vytvořené zálohy lze přehledně zobrazit jak z PC1, tak ze ZÁLOHOVACÍHO PC2 jednoduchým příkazem:

```
rdiff-backup -l student@158.196.142.77::/cesta_k_zaloze/
```

Výstup výše zmíněného příkazu zobrazuje obrázek 4.3, můžeme z něj vyčíst, ze kterého dne je dostupná zrcadlová kopie v nekomprimované podobě (current mirror) a jak staré jsou inkrementální zálohy. Patrné je také to, že inkrementální zálohy tvoří řadu na sebe navazujících záloh, která by neustále narůstala a je omezena pouze počtem záloh, které chceme uchovávat. Nejsou tedy doplněny o více plných záloh.



Obrázek 4.3: Přehled záloh vytvořených nástrojem rdiff-backup

DUPLICITY

Dalším z testovaných nástrojů byl DUPLICITY. Při testování došlo ke komplikacím, jelikož nešly odstraňovat staré zálohy. V důsledku toho by docházelo k rychlému vyčerpání kapacity USB flash disku na, který se zálohovalo.

Při použití `remove-all-but-n-full` nebo `remove-older-than` byly odstraněny pouze manifest soubory, ale archívy s daty zůstaly. K této situaci docházelo v případech, kdy byl pro přenos zálohy využit protokol SCP. Řešením tohoto problému bylo místo SCP využít rsync a vše již fungovalo podle očekávání.

V následující ukázce části statistiky po dokončení první zálohy nástrojem duplicity můžeme vidět velikost zdrojového adresáře (SourceFileSize). Ta je tentokrát uvedena včetně těch souborů a adresářů, které mají být ze zálohování vynechány. Dále velikost nových souboru (NewFileSize) a velikost nových dat v adresáři, kam se záloha ukládá. Kompletní statistika je k dispozici v příloze H.

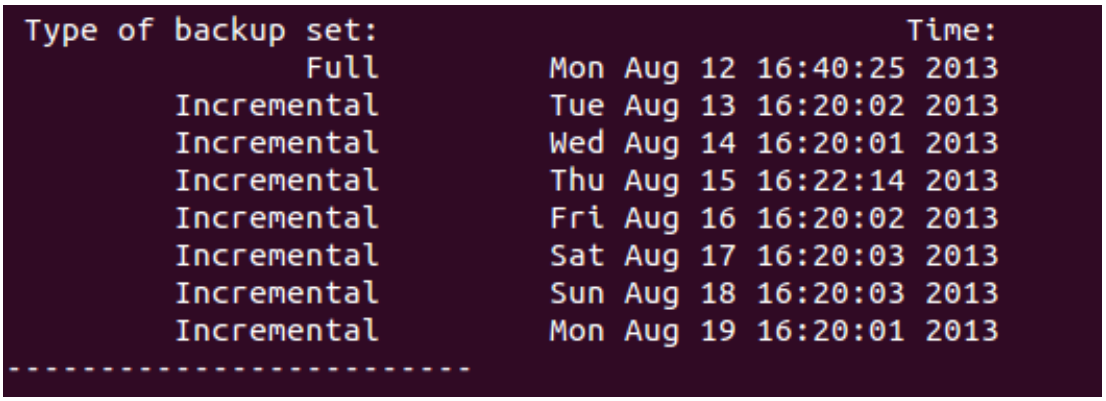
```
SourceFileSize 294368207 (281 MB)

NewFileSize 294368207 (281 MB)

TotalDestinationSizeChange 187082255 (178 MB)
```

Statistika po dokončení inkrementální zálohy opět přehledně zobrazí novou celkovou velikost zdrojového adresáře včetně souborů, které nemají být zálohovány. Velikost nových souborů ve zdrojovém adresáři a také velikost nové inkrementální zálohy. Kompletní statistika je k dispozici v příloze I. Ze statistik opět vyplývá, že byla provedena inkrementální záloha což, splnilo očekávání.

```
SourceFileSize 344404023 (328 MB)

NewFileSize 49892854 (47.6 MB)

TotalDestinationSizeChange 56382092 (53.8 MB)
```

Na obrázku 4.4 je zobrazen přehledný stav záloh po dokončení sedmidenního cyklu zálohování. Snadno lze rozlišit, jestli se jedná o plnou nebo inkrementální zálohu a dny, ve kterých byly vytvořeny. Bohužel nezobrazuje velikosti jednotlivých záloh. Takový to přehled lze zobrazit následujícím příkazem.

```
duplicity collection-status rsync://student@158.196.142.77//
cesta_k_zaloze /
```



Obrázek 4.4: Přehled záloh vytvořených nástrojem duplicity

HDUP2

Poslední z testovaných nástrojů je hdup2. Tento nástroj jsem zvolil proto, aby mezi testovanými nástroji byl zahrnut i už hotové řešení, které stačí pouze správně nastavit. Během testování vyšlo najevo, že neobsahuje volbu include a také nelze vynechávat pouze určité typy souborů.

Statistiky po ukončení měsíční zálohy nástrojem hdup2 neuvádí velikost zdrojového adresáře, ale pouze velikost vytvořeného komprimovaného archívu tar.bz2. Dalšími důležitými údaji, které jsou uvedeny, je jméno profilu, pod kterým byla záloha vytvořena, o jaký typ zálohy se jednalo (měsíční, denní, týdenní) a jestli bylo zapnuto šifrování, viz následující ukázka části statistiky.

```
Hdup version.:  2.0.14

Host.........:  student

Date.........:  2013-04-07

Scheme.......:  monthly

Archive......:  student.2013-04-07.monthly.tar.bz2

Encryption...:  no

Bytes written:  200.2M

Elapsed......:  0:05:51

Status.......:  successfully performed backup
```

Vytvořená záloha má velikost 200,2 MB, což je více než bylo u předchozích nástrojů, jelikož je zálohován i soubor s příponou .avi, který byl u ostatních nástrojů vynechán a podadresář a celý adresář `/Pictures /`. Zároveň je, ale provedena komprimace. Velikost vytvořené zálohy odpovídá velikosti, jakou by měl adresář `/student/` v případě, že bychom provedli jeho archivaci nástrojem tar a zároveň komprimaci nástrojem bzip2 bez adresáře `/Downloads/` (ten je ze zálohy vynechán), protože záloha vytvořená nástrojem hdup2 je komprimována.

Následující ukázka je část statistiky po dokončení zálohy týdenního typu. Všechny informace zůstaly stejné jako v předchozím případě, pouze velikost zapsaných dat je rozdílná. Podle předpokladů se zálohovaly pouze změny od předchozí měsíční zálohy. V případě, že by byla provedena další týdenní záloha a mezi tím by nebyla vytvořena nova měsíční záloha, opět by se zálohovalo vše od původní měsíční zálohy. Kompletní statistika je přiložena jako příloha J.

```
Host.........:  student

Scheme.......:  weekly

Bytes written:  47.4M
```

Posledním typem zálohy, kterou hdup2 nabízí, je denní. Ta zálohuje změny od poslední týdenní zálohy, jestliže nedojde k žádným změnám od poslední týdenní zálohy, pak nebude nic zálohováno, viz příloha K, ve které statistika popisující tento případ. V případě hdup2 se také podařilo ověřit správnou funkci navrženého řešení.

5 Závěr

Cílem této bakalářské práce bylo navrhnout možná řešení zálohování dat v síti pomocí open source nástrojů, tato řešení následně ověřit v laboratorních podmínkách a popsat jejich výhody a nevýhody. V jednotlivých kapitolách práce byly vymezené cíle postupně splněny. Teoretická část si kladla za cíl dostatečně objasnit důležitost zálohování dat, a podstatná část se také zabývala popsáním používaných technologií pro ukládání dat v síti, jako jsou DAS, NAS a SAN. Velikou oblibu si v současné době získává NAS díky nízkým pořizovacím cenám a možnosti zapojení do již vybudované LAN sítě. Zároveň se i technologie SAN stává dostupnější díky protokolu iSCSI.

Stěžejní částí této práce bylo navrhnout možná řešení zálohování dat v síti. K tomuto účelu jsem zvolil čtyři open source nástroje, dostupné pod OS Linux. Z široké nabídky jsem vybral rsync, rdiff-backup, duplicity a hdup2. Největší pozornost byla věnována nástroji rsync, jelikož ten tvoří základ pro mnoho dalších zálohovacích nástrojů. Řešení navržené tímto nástrojem vytváří přehlednou strukturu inkrementálních záloh, které odpovídají sedmi dnům v týdnu, a rovněž udržuje staré zálohy v komprimované podobě. Výhodou tohoto řešení je to, že sedm posledních záloh je dostupných v otevřené podobě a můžeme snadno obnovovat konkrétní soubory. Slabinou rsync je to, že kontrolní součty, které používá pro určení změn, si nikam neukládá. Další z nástrojů byl rdiff-backup, obdobně jako rsync si neukládá kontrolní součty již vytvořených záloh. Výhodou tohoto nástroje je snadné ovládání i přesto, že je to nástroj určený pro terminál, jinak ovšem oproti ostatním nástrojům nevyniká. Jako nejvšestrannější z vybraných nástrojů a dle mého názoru také nejlepší z těch, se kterými jsem v průběhu této práce pracoval, byl duplicity. Jako jediný si ukládá kontrolní součty vytvořených záloh, navíc na rozdíl od rdiff-backup a hdup2 nemusí být nainstalován i na vzdáleném serveru, kam je záloha ukládána. Další z jeho výhod je možnost šifrovat zálohy jak symetrickou, tak asymetrickou šifrou. Poslední řešení bylo navrženo pomocí hdup2. V průběhu práce s tímto nástrojem vyšlo najevo, že neumožňuje ze zálohy vynechávat pouze určité typy souborů. Vzhledem k tomu by bylo lepší volbou využít duplicity.

Přínosem této bakalářské práce pro mne bylo seznámení se s různými zálohovacími nástroji pro OS Linux a velké rozšíření znalostí práce s tímto OS. Znalosti získané při psaní práce považuji za přínos a užitečnou praxi do budoucna. Přínos mé bakalářské práce v oblasti zálohování dat vidím v tom, že ukázala různé možnosti zálohování dat v síti využitím open source nástrojů v OS Linux. Rovněž poukázala na to, že zálohování lze provádět kvalitně i pomocí open source nástrojů. Navržená řešení by šla dále rozšiřovat, což se týká hlavně rsync. Mnou navržené řešení by mohlo být v budoucnu doplněno o šifrování archívů, které moje řešení vytváří. Tím by se zajistila vyšší bezpečnost vytvořených záloh.

Použitá literatura

[1] Správa linuxového serveru: Úvod do zálohování - Linux E X P R E S. *Linux E XPRES* [online]. 2010 [cit. 2013-03-05]. Dostupné z: http://www.linuxexpres.cz/praxe/sprava-linuxoveho-serveru-uvod-do-zalohovani

[2] PRESTON, W. Curtis. *Backup & Recovery: Inexpensive Backup Solutions for Open Systems*. United States of America: O'Reilly Media, 2007. ISBN ISBN 978-0-596-10246-3. Dostupné z: http://books.google.cz/books?id=6-w4fXbBInoC&printsec=frontcover&hl=cs&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

[3] SOUBOR AUTORŮ. *Linux: Dokumentační projekt*. 4. aktualizované vydání. Brno: Computer Press a. s, 2007. ISBN ISBN 978-80-251-1525-1.

[4] NELSON, Steven. *Pro Data Backup and Recovery*. New York: Apress, 2011. ISBN 978-1-4302-2662-8.

[5] Backup - Wikipedia, the free encyclopedia. *Wikipedia, the free encyclopedia* [online]. 2013 [cit. 2013-02-28]. Dostupné z: http://en.wikipedia.org/wiki/Backup

[6] Direct-attached storage - Wikipedia, the free encyclopedia. *Wikipedia, the free encyclopedia* [online]. 2013 [cit. 2013-03-10]. Dostupné z: http://en.wikipedia.org/wiki/Direct-attached_storage

[7] Network-attached storage - Wikipedia, the free encyclopedia. *Wikipedia, the free encyclopedia* [online]. 2013 [cit. 2013-03-10]. Dostupné z: http://en.wikipedia.org/wiki/Network-attached_storage#NAS_vs._DAS

[8] Storage area network - Wikipedia, the free encyclopedia. *Wikipedia, the free encyclopedia* [online]. 2013 [cit. 2013-03-10]. Dostupné z: http://en.wikipedia.org/wiki/Storage_area_network

[9] Ukládání dat SAN - Fibre Channel | Technické okénko | Technická podpora a další důležitá dokumentace | Vahal s.r.o. - hardware a software. *Vahal s.r.o. - hardware a software* [online]. 2009 [cit. 2013-03-10]. Dostupné z: http://www.vahal.cz/cz/podpora/technicke-okenko/ukladani-dat-san-fc.html

[10] Rsync. *Rsync* [online]. 2011 [cit. 2013-03-14]. Dostupné z: http://rsync.samba.org/ftp/rsync/rsync.html

[11] Šifrované inkrementální zálohy s Duplicity - Root.cz. *Root.cz - informace nejen ze světa Linuxu* [online]. 2011 [cit. 2013-03-19]. Dostupné z: http://www.root.cz/clanky/sifrovane-inkrementalni-zalohy-s-duplicity/

[12] Magnetic tape - Wikipedia, the free encyclopedia. *Wikipedia, the free encyclopedia* [online]. 2013 [cit. 2013-04-28]. Dostupné z: http://en.wikipedia.org/wiki/Magnetic_tape

[13] RAID jako RAID – díl 1. Úvod - Jens.cz. *Jens.cz - zápisník programátora* [online]. 2009 [cit. 2013-04-28]. Dostupné z: http://www.jens.cz/neni-raid-jako-raid-dil-1-uvod/

[14] Standard RAID levels - Wikipedia, the free encyclopedia. *Wikipedia, the free encyclopedia* [online]. 2013 [cit. 2013-04-28]. Dostupné z: http://en.wikipedia.org/wiki/Raid_6#RAID_6

[15] Cron – správca úloh - Linux E X P R E. *Linux E X P R E S* [online]. 2007 [cit. 2013-04-28]. Dostupné z: http://www.linuxexpres.cz/praxe/cron-spravca-uloh

[16] RDIFF-BACKUP. *Welcome [Savannah]* [online]. 2009 [cit. 2013-04-30]. Dostupné z: http://www.nongnu.org/rdiff-backup/rdiff-backup.1.html

[17] Backup with hdup - ArchWiki. *Arch Linux* [online]. 2009 [cit. 2013-04-30]. Dostupné z: https://wiki.archlinux.org/index.php/Backup_with_hdup

[18] Rsync FAQ. *Rsync* [online]. 2011 [cit. 2013-04-30]. Dostupné z: http://rsync.samba.org/FAQ.html

[19] Data v péči MHM - Fenomén iSCSI. *Data v péči MHM - číslo 29/2012* [online]. 2007 [cit. 2013-05-02]. Dostupné z: http://www.datavpeci.cz/webdvp.nsf/0/07ABA713F282B84CC125750C00471D9D

Seznam obrázků

Seznam příloh

Příloha.A: Zálohovací skript pro rsync

```
#!/bin/sh
##########
#první část skriptu, která vytváří inkrementální zálohy
#Získání dnešního data
TODAY=`date -I`
#Získání včerejšího data
YESTERDAY=`date -I -d "1 day ago"`
#Zdrojový adresář, který zálohujeme
SOURCE="student@158.196.142.76:/home/student/"
#Cílový adresář, kde bude záloha ukládána
TARGET="/media/backup/incremental/$TODAY"
#Adresář, s kterým bude záloha porovnávána pro vytvoření
#inkrementální
LINK="/media/backup/incremental/$YESTERDAY"
#Přepínače pro rsync
OPTIONS="-ahze ssh --exclude=*.avi --include=/Pictures/Important/ --
exclude=/Pictures/* --exclude=/Downloads/ --delete --stats --link-
dest=$LINK"
#Kontrola existence adresáře incremental
if [ -e /media/backup/incremental/ ]
then
echo "Directory incremental OK."
else
mkdir /media/backup/incremental/
fi
# Kontrola existence adresáře archive
if [ -e /media/backup/archive/ ]
then
echo "Directory archive OK."
else
mkdir /media/backup/archive/
fi
echo "Backup progress, please wait."
```

```
#Samotné spuštění zálohování s parametry OPTIONS, zdrojem SOURCE a
#cílem TARGET

rsync $OPTIONS $SOURCE $TARGET

#Získání 7 dnů starého data

DAY7=`date -I -d "7 days ago"`

#Smazání starých záloh

if [ -d /media/backup/incremental/$DAY7 ]

then

echo "Deleting old backup, please wait."

rm -R /media/backup/incremental/$DAY7

fi

##########

#druhá část skriptu, která vytváří komprimované archívy všech
#týdenních inrementálních záloh

##########

#Získání aktuálního dne. Jestliže je sobota, pak provede archivaci
#inkrementálních záloh.

DENVTYDNU=`date +%u`

if [ $DENVTYDNU = 6 ]

then

echo "Archiving backup, please wait."

tar -cvzf /media/backup/archive/$TODAY.tar.gz
/media/backup/incremental/

fi

#Určení počtu souboru v adresáři archive, poté najde nejstarší
#soubor a ten smaže, jestliže je v adresáři více než 2 soubory

COUNT=`find /media/backup/archive/ -type f | wc -l`

DEL_ARCH=`ls /media/backup/archive/ -t | tail -n 1`

if [ $COUNT -gt 7 ]

then

echo "Deleting old backup, please wait."

rm -R /media/backup/archive/$DEL_ARCH

fi

echo "Backup end."
```

Příloha.B: Zálohovací skript pro rdiff-backup

```
#!/bin/sh
#Zdrojový adresář, který zálohujeme
SOURCE="/home/student/"
#Cílový adresář, kde bude záloha ukládána
TARGET=student@158.196.142.77::/media/backup/
#Přepínače pro rdiff-backup
OTIONS="--force -v3 --print-statistics --exclude
/home/student/**.avi --include /home/student/Pictures/Important --
exclude /home/student/Pictures/ --exclude /home/student/Downloads/"

#Přepínače pro rdiff-backup k mazání
OTIONS_RM="--remove-older-than 1M --force"
#Spuštění zálohování s parametry OPTIONS, zdrojem SOURCE a cílem
#TARGET
echo "Backup progress, please wait."
rdiff-backup $OTIONS $SOURCE $TARGET
echo "Backup end."
#Spuštění mazání záloh s parametry OPTIONS_RM a cílem TARGET
echo "Deleting old backup, please wait."
rdiff-backup $OTIONS_RM $TARGET
```

Příloha.C: Zálohovací skript pro duplicity

```
#!/bin/bash
#Zdrojový adresář, který zálohujeme
SOURCE="/home/student/"
#Cílový adresář, kde bude záloha ukládána
#pro přenos je využit rsync
TARGET="rsync://student@158.196.142.77//media/backup/"
#Heslo, které je využito pro šifrování. Nejedná se o heslo pro
#připojení na vzdáleny stroj
export PASSPHRASE='tajneheslo'
```

```
#Přepínače pro duplicity
OPTIONS="--exclude /home/student/**.avi --include
/home/student/Pictures/Important --exclude /home/student/Pictures --
exclude /home/student/Downloads --full-if-older-than 7D"

#Přepínače pro duplicity k mazání starých záloh. Ponechá posledních
5 plných a na ně navazující inkrementální
OPTIONS_RM="remove-all-but-n-full 3 --force"

#Spuštění zálohování s parametry OPTIONS, zdrojem SOURCE a cílem
TARGET
duplicity $OPTIONS $SOURCE $TARGET

#Spuštění mazání záloh s parametry OPTIONS_RM a cílem TARGET
duplicity $OPTIONS_RM $TARGET
```

Příloha.D: Generování RSA klíčů

RSA klíče vygenerujeme příkazem:

```
ssh-keygen –t rsa
```

Následně veřejný klíč překopírujeme na vzdálený stroj, ke kterému chceme přistupovat:

```
scp .ssh/id_rsa.pub student@158.196.142.76:~
```

Po zkopírování klíče se připojíme na vzdálený stroj (zatím s heslem) a provedeme poslední dva příkazy:

```
ssh student@158.196.142.76
cd $HOME
cat id_rsa.pub >>.ssh/authorized_keys
```

Odpojíme se od vzdáleného stroje:

```
EXIT
```

A ověříme ssh přístup bez zadání hesla:

```
ssh student@158.196.142.76
```

a připojení by mělo proběhnout bez dotazu na heslo uživatele.

Příloha.E: Výpis po ukončení inkrementální zálohy skriptem rsync

```
Number of files: 1483
Number of files transferred: 17
Total file size: 230.07M bytes
Total transferred file size: 49.42M bytes
Literal data: 48.63M bytes
Matched data: 797.52K bytes
File list size: 37.96K
File list generation time: 0.002 seconds
File list transfer time: 0.000 seconds
Total bytes sent: 8.88K
Total bytes received: 47.96M

sent 8.88K bytes  received 47.96M bytes  1.28M bytes/sec
total size is 230.07M  speedup is 5.00
```

Příloha.F: Statistika po ukončení první zálohy nástrojem rdiff-backup

```
--------------[ Session statistics ]--------------
StartTime 1366873543.00 (Thu Apr 25 09:05:43 2013)
EndTime 1366873657.86 (Thu Apr 25 09:07:37 2013)
ElapsedTime 114.86 (1 minute 54.86 seconds)
SourceFiles 1475
SourceFileSize 191763792 (183 MB)
MirrorFiles 1
MirrorFileSize 0 (0 bytes)
NewFiles 1474
NewFileSize 191763792 (183 MB)
DeletedFiles 0
DeletedFileSize 0 (0 bytes)
ChangedFiles 1
ChangedSourceSize 0 (0 bytes)
ChangedMirrorSize 0 (0 bytes)
IncrementFiles 0
IncrementFileSize 0 (0 bytes)
TotalDestinationSizeChange 191763792 (183 MB)
Errors 0
-------------------------------------------------
```

Příloha.G: Statistika po ukončení inkrementální zálohy nástrojem rdiff-backup

```
-------------[ Session statistics ]-------------
StartTime 1366986047.00 (Fri Apr 26 16:20:47 2013)
EndTime 1366986119.25 (Fri Apr 26 16:21:59 2013)
ElapsedTime 72.25 (1 minute 12.25 seconds)
SourceFiles 1480
SourceFileSize 240019180 (229 MB)
MirrorFiles 1475
MirrorFileSize 191763792 (183 MB)
NewFiles 6
NewFileSize 48111507 (45.9 MB)
DeletedFiles 1
DeletedFileSize 32768 (32.0 KB)
ChangedFiles 34
ChangedSourceSize 59444268 (56.7 MB)
ChangedMirrorSize 59267619 (56.5 MB)
IncrementFiles 41
IncrementFileSize 3246666 (3.10 MB)
TotalDestinationSizeChange 51502054 (49.1 MB)
Errors 0
-------------------------------------------------
```

Příloha.H: Statistika po ukončení první zálohy nástrojem duplicity

```
--------------[ Backup Statistics ]--------------
StartTime 1366873183.50 (Thu Apr 25 08:59:43 2013)
EndTime 1366873280.51 (Thu Apr 25 09:01:20 2013)
ElapsedTime 97.02 (1 minute 37.02 seconds)
SourceFiles 1578
SourceFileSize 294368207 (281 MB)
NewFiles 1578
NewFileSize 294368207 (281 MB)
DeletedFiles 0
ChangedFiles 0
ChangedFileSize 0 (0 bytes)
ChangedDeltaSize 0 (0 bytes)
DeltaEntries 1578
RawDeltaSize 293150803 (280 MB)
TotalDestinationSizeChange 187082255 (178 MB)
Errors 0
-------------------------------------------------
```

Příloha.I: Statistika po dokončení inkrementální zálohy nástrojem duplicity

```
-------------[ Backup Statistics ]--------------
StartTime 1366986336.73 (Fri Apr 26 16:25:36 2013)
EndTime 1366986375.88 (Fri Apr 26 16:26:15 2013)
ElapsedTime 39.15 (39.15 seconds)
SourceFiles 1596
SourceFileSize 344404023 (328 MB)
NewFiles 32
NewFileSize 49892854 (47.6 MB)
DeletedFiles 1
ChangedFiles 18
ChangedFileSize 59464910 (56.7 MB)
ChangedDeltaSize 0 (0 bytes)
DeltaEntries 51
RawDeltaSize 63608749 (60.7 MB)
TotalDestinationSizeChange 56382092 (53.8 MB)
Errors 0
-------------------------------------------------
```

Příloha.J: Statistika po dokončení týdenní zálohy nástrojem hdup2

```
Hdup version.:  2.0.14
Host.........:  student
Date.........:  2013-04-14
Scheme.......:  weekly
Archive......:  student.2013-04-14.weekly.tar.bz2
Encryption...:  no
Bytes written:  47.4M
Elapsed......:  0:02:24
Status.......:  successfully performed backup
```

Příloha.K: Statistika po dokončení denní zálohy nástrojem hdup2

```
Hdup version.:  2.0.14
Host.........:  student
Date.........:  2013-04-15
Scheme.......:  daily
Archive......:  student.2013-04-15.daily.tar.bz2
Encryption...:  no
Bytes written:  7k
Elapsed......:  0:00:00
Status.......:  successfully performed backup
```

Příloha.L: Konfigurační soubor hdup.conf

Tento konfigurační soubor odpovídá nastavení lokálního PC, který má být zálohován. Na cílovém stroji, kam je záloha ukládána je totožný konfigurační soubor pouze je změněna hodnota `allow remote = no` na `allow remote = yes`. Proto zde není uveden konfigurační soubor pro lokální i vzdálený stroj zvlášť.

```
# everything put under [global] is "inherited" by
# all the other hosts defined in this config file
[global]
# where to put the tar archives
archive dir = /media/backup/
# use the normal date
date spec = iso
# try to figure out the current scheme, and perform the
# correct backup
always backup = on
# skip the archive dir from the backup
skip = on
# restore option: disallow restoring to /
force = no
# overwrite existing archives in 'archive dir'
overwrite = on
# ssh options
proto = /usr/bin/ssh
```

```
proto option =  -q -oProtocol=2
# chown the archives to this user
user = student
# compression options
compression = bzip
compression level = 6
# is such a file is found, exclude that directory
nobackup = .nobackup
# give tar some extra options, not needed
# tar option =
# my own host
[student]
# what to backup, separate with ,.
# For directories add closing slash, like /home/
dir = /home/student/
# don't include theses directories
exclude = /Downloads/
# if we want to split it (to fit a CD)
#chunk size = 640m
# enable for nagging
#postrun = /etc/hdup/postrun-warn-user %s
# log to syslog
log = yes
exclude = /var/.*/docs/
one filesystem = no
compression = gzip
#key = key2
#algorithm = tripledes
allow remote = no
free = 20m
#prerun = echo %c %e
remote hdup = /mnt/key/src/hdup2
remote hdup option = -c /mnt/key/hdup2.conf
```